Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки Программная инженерия

РАБОТА ПРОВЕРЕНА
Рецензент,

_____
«____»_____2018 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н.,
доцент
_____/А.А.Замышляева
«____»_____2018 г.

Разработка мобильного приложения для помощи сбора личных вещей

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–09.03.04.2018.92. ПЗ ВКР

Руководитель работы, к.э.н., доцент
_____/Д.А. Дрозин
«____»_____2018 г.
Автор работы
Студентка группы ЕТ-414
_____/ А.Н. Ершова
«____»_____2018 г.
Нормоконтролер, к.т.н., доцент
_____/Т.Ю.Оленчикова
«____»_____2018 г.

Челябинск 2018

АННОТАЦИЯ

Ершова А.Н. Разработка мобильного приложения для помощи сбора личных вещей. - Челябинск: ЮУрГУ, ЕТ-414, 44 с, 33 ил., библиогр. список – 16 наим.

В представленной выпускной квалификационной работе проводится реализация мобильного приложения для операционной системы Android. Данное приложение призвано помочь путешественникам и деловым людям при сборе личных вещей.

В работе приведен обзор существующих решений, выявлены основные требования к системе. Спроектировано и реализовано программное решение. Включены диаграммы классов, компонентов, деятельности и вариантов использования. В приложениях приведены код и описание приложения.

Преимуществами данного приложения, которые позволят ей занять свою нищу на рынке, можно назвать простой и эргономичный дизайн, а так же отсутствие излишнего функционала.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Человек в современном мире отличается высокой степенью мобильности. Рабочие командировки, туризм, поездки с друзьями и родственниками, иногороднее и зарубежное образование обуславливают данную тенденцию. Любое путешествие предваряется времязатратным и напряженным сбором личных вещей. Основной проблемой сложившейся ситуации является сложность удержания в памяти списка необходимых для поездки предметов, а также возможность забыть некоторые вещи. Как следствие, могут появиться финансовые издержки на приобретение забытых вещей в дороге, потеря времени на возвращение в пункт отправления, излишний стресс и тревога, в крайних случаях отмена всего мероприятия.

Цель выпускной квалификационной работы заключается в разработке мобильного приложения, способствующего облегчению сбора личных вещей для человека перед предстоящим путешествием и позволяющего избежать перечисленых выше проблем.

Основными пользователями разрабатываемого приложения являются активные, часто путешествующие люди, а также те, кто имеют работу, предусматривающую частые командировки.

Для удовлетворения потребностей пользователей, приложение должно предоставлять им следующшие возможности:

1) создание списков личных вещей с указанием наименования списка, места назначения, даты поездки;
2) возможность просмотра необходимого списка и реализация отметок для уже собранных вещей в списке;
3) редактирование списка, а именно: добавление необходимых пунктов и удаление ненужных, а также редактирование данных о поездке;
4) удаление списков.

1 МОБИЛЬНЫЕ ПРИЛОЖЕНИЯ СБОРА ЛИЧНЫХ ВЕЩЕЙ ПУТЕШЕСТВЕННИКА

1.1 «PackPoint»

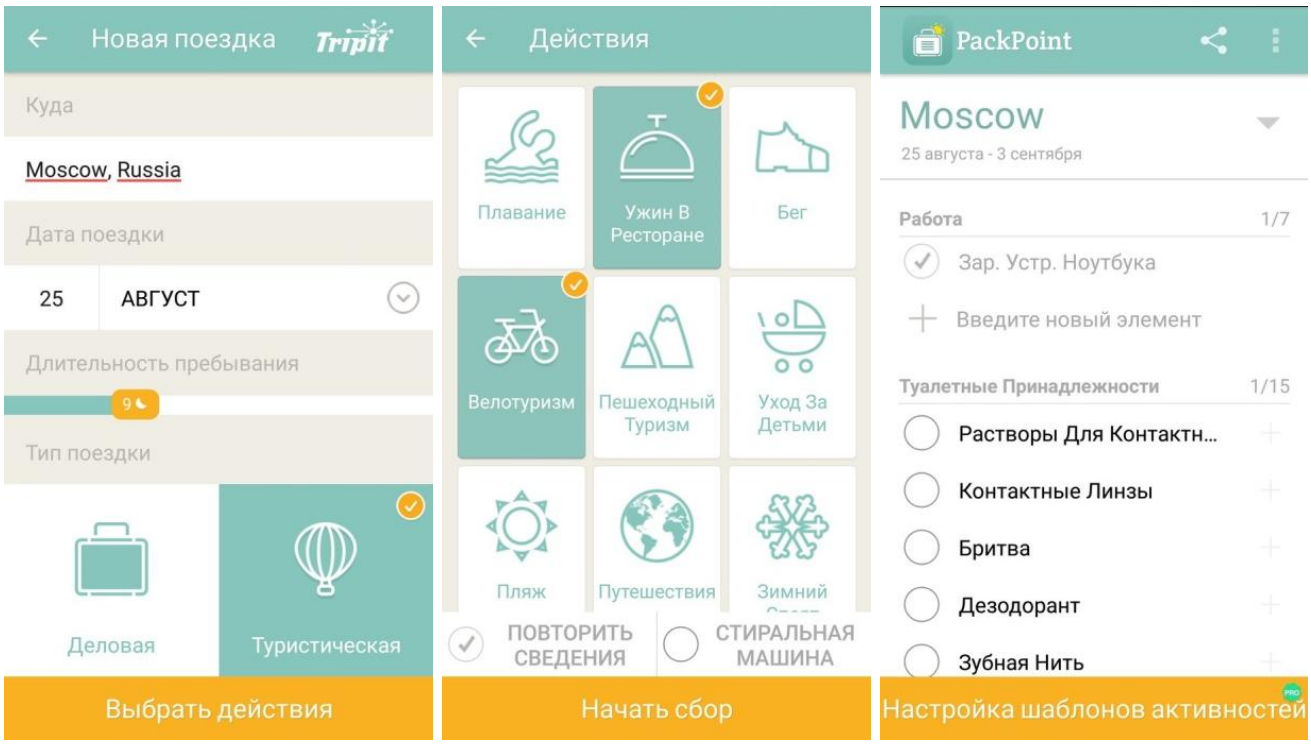
«PackPoint» – это мобильное приложение для Android и iOS. Рассмотрим принципы работы с данным приложением [1].

На начальном экране приложения расположен раскрывающийся список со всеми созданными пользователем ранее списками вещей. Ниже помещена краткая информация о прогноз погоды в месте назначения выбранного списка, и далее располагается сами пункты.

Для создания нового списка вещей, как показано на рисунке 1.1, необходимо указать пункт назначения, дату поездки, ее длительность и тип. В типе предложены два варианта: деловая и туристическая поездка. Можно выбрать два типа одновременно. Далее предлагается выбрать действия, своего рода категории, по которым будет строиться подборка пунктов списка для пользователя, такие как: плавание, ужин в ресторане, бег, туризм, зимний спорт, фотография и др. Если пропустить данный этап, то в сформированном списке будут указаны только предметы первой необходимости и туалетные принадлежности. Также при оформлении платной подписки появляется возможность создавать пользовательские действия. После указания всех необходимых категорий формируется список на основании введенных данных. Кроме того, перед составлением списка, данный программный продукт получает из сети данные о прогнозе погоды на указанный период поездки, что влияет на состав предложенного списка вещей.

Чтобы удалить ненужный пункт достаточно движения пальца влево или вправо. Для добавления собственного пункта в конце каждой категории имеется строка для ввода. Также можно указать количество вещей и отметить пункт как взятый (около него появится галочка, и он опустится в низ списка).

В настройках приложения можно указать пол пользователя (это будет отражаться на составе формируемых списков), единицы измерения температуры, просмотреть архив поездок, а при оформлении платной подписки функционал продукта можно расширить еще некоторыми возможностями, например, получение уведомлений о предстоящих поездках.



а) б) в)

Рисунок 1.1 – Мобильное приложение «PackPoint»: а) ввод основных данных о списке, б) выбор действий, в) начальный экран приложения с созданными списками

Достоинства:

1) гибкое управление созданием списка (свободное добавление собственных пунктов, а также удаление их из предложенного списка);
2) возможность поделиться списком с другими людьми (через почту и мессенджеры);
3) учет пола пользователя, погодных условий, типа и длительности поездки;
4) архив поездок.

Недостатки:

1) приложение работает только при наличии Интернета;
2) возможность создания собственных видов деятельности является платной;
3) пункты в шаблонных списках субъективны, сами списки длинные, придется потратить много времени на удаление неподходящих пунктов;
4) невозможно определить собственный порядок групп, по которым распределены пункты;
5) невозможно добавить пункт вне группы, а функция создания собственной группы является платной.

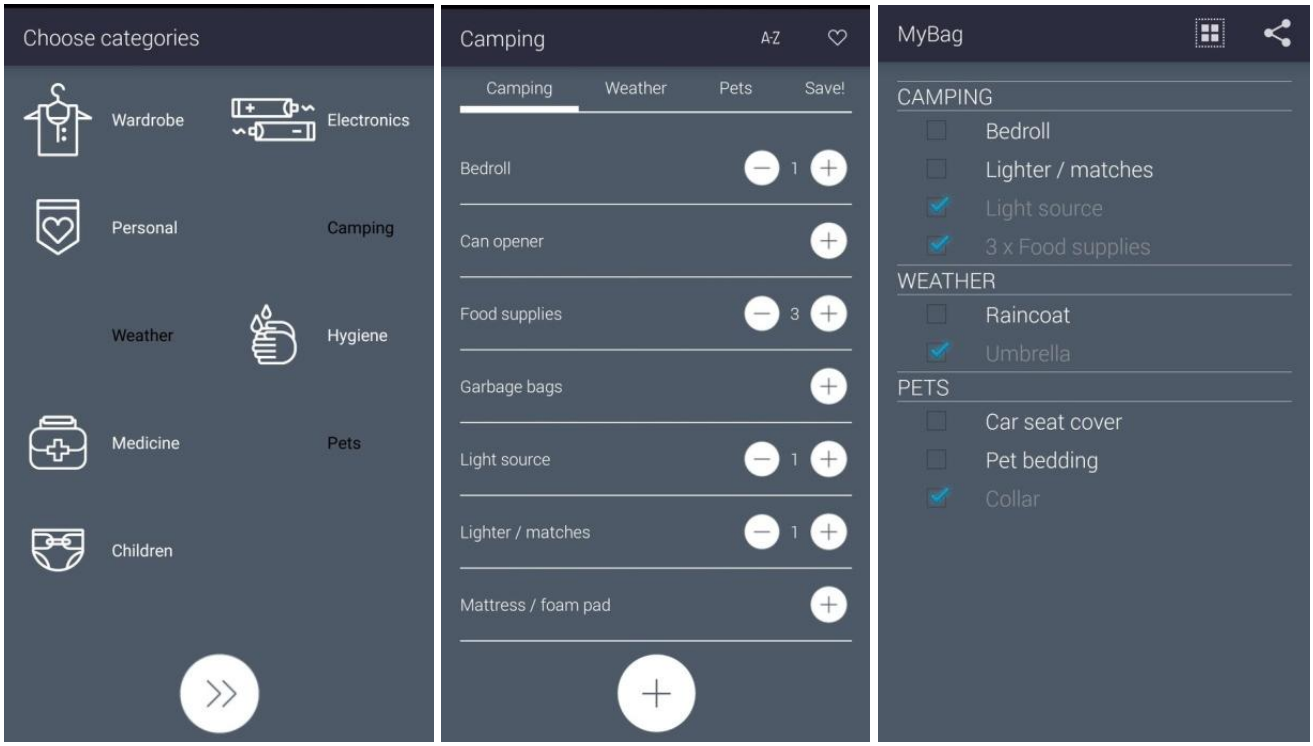
1.2 «PackMeApp Packing List»

PackMeApp Packing List – еще одно мобильное приложение для Android, представленное на рисунке 1.2 [1].

На начальном экране приложения в виде списка располагаются все ранее созданные списки вещей, с расположенными рядом иконками редактирования и удаления.

Для того чтобы создать новый список вещей необходимо указать категории вещей из небольшого числа вариантов. Это, например, гардероб, дети, медицина, животные, погода и др. Далее под каждой выбранной категорией будет предложен шаблонный список вещей, в котором нужно выбрать интересующие пункты и при необходимости указать их количество. Также можно создать собственные пункты с указанием важности, которая будет использована впоследствии (в приложении реализованы два вида сортировки пунктов в списке: сортировка в алфавитном порядке и сортировка по важности). Можно удалить шаблонный пункт, чтобы в дальнейшем при формировании списка он не предлагался приложением. В настройках приложения на начальном экране можно найти удаленные шаблонные пункты и восстановить их. После этого формируется и сохраняется новый список вещей.

При просмотре списка пункты будут разделены на категории. Можно отметить пункт как упакованный (около него появиться галочка, цвет шрифта потемнеет, и пункт опуститься в конец списка своей категории).



а) б) в)

Рисунок 1.2 – Мобильное приложение «PackMeApp Packing List»: а) выбор категорий, б) выбор пунктов по категориям, в) просмотр списка

Достоинства:

1) наличие категорий вещей для упрощения создания списка;
2) простой и понятный дизайн;

3) возможность поделиться списком с другими людьми (через почту и мессенджеры);
4) возможность редактирования шаблонных списков: удаление ненужных пунктов из базы данных приложения, создание собственных пунктов. При добавлении пункта предусмотрена функция установки приоритета пункта. При создании нового списка пункты, у которых приоритет выше, устанавливаются в начало списка.

Недостатки:

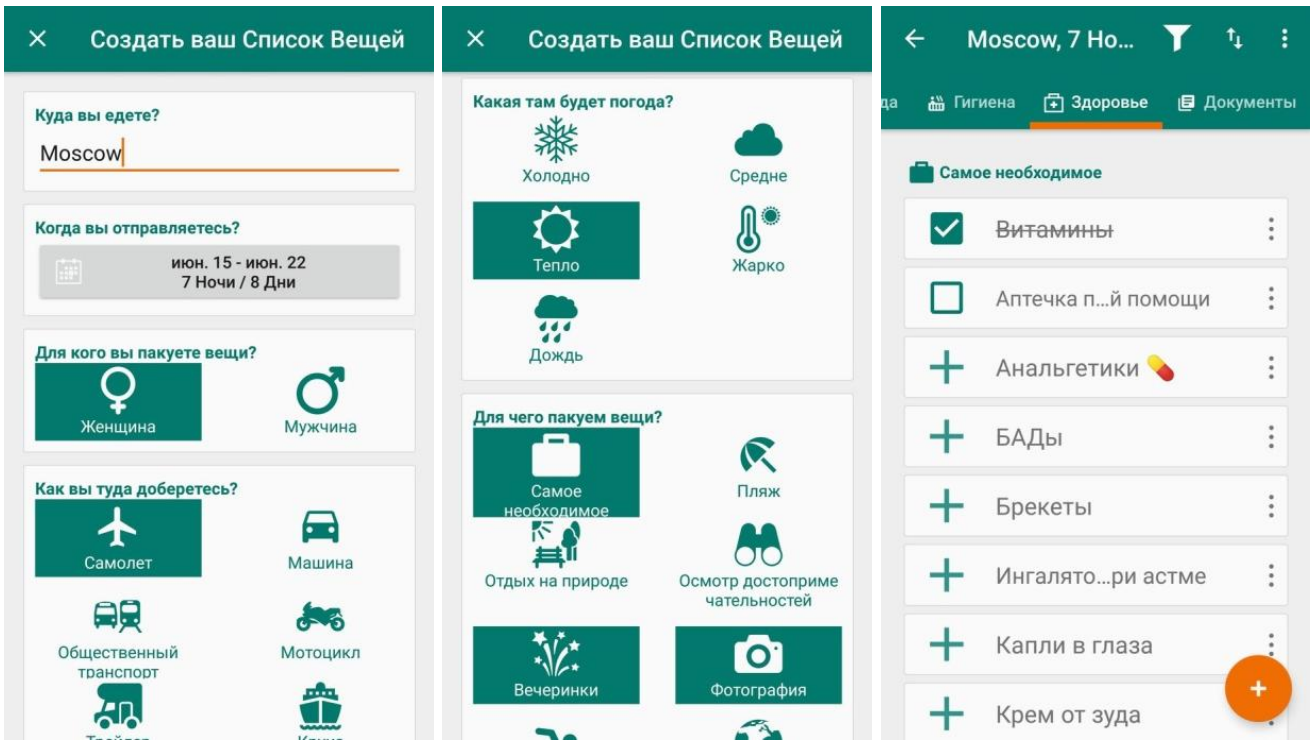
1) отсутствие возможности создать список "с нуля" (необходимость выбора хотя бы одной категории);
2) малое количество категорий (всего девять наименований);
3) отсутствие функции создания пользовательской категории;
4) непроработанный пользовательский интерфейс (если установлена серая тема, то у выбранной категории иконка сливается с фоном, при выборе категории нажатие кнопки Back скрывает приложение, а не возвращает на главное окно и т.п.).

1.3 «PackKing»

Мобильное приложение для Android [1]. На начальном экране расположены все списки вещей, созданные пользователем, с указанием периода поездки, места назначения и индикацией, показывающей количество отмеченных пунктов из всех пунктов поштучно, а также в процентном соотношении (см. рисунок 1.3).

Для создания нового списка вещей необходимо указать место назначения, временной период, пол пользователя (опционально), тип транспорта, на котором предполагается добраться до пункта назначения (опционально), предполагаемую погоду в месте пребывания, темы вещей, например, вечеринки, пеший туризм, фитнес, поход с палатками, рыбалка, гольф, путешествия на лодке и др. В приложении можно создавать пользовательские темы. После этого формируется предлагаемый приложением список вещей. Пункты распределены по категориям, в которых еще есть деление по темам. Примеры категорий: одежда, гигиена, здоровье, документы, устройства и проч. Каждый предложенный пункт можно добавить в окончательный список, удалить из шаблонного списка, переименовать, выбрать количество данного пункта. Также любой пункт можно добавить в специальный список, называемый «Shopping list». Для удобства просмотра и поиска вещей в приложении реализованы различные фильтры. Например, фильтр, показывающий отмеченные пункты или наоборот. Также приложение имеет подсказки по сбору вещей в виде подборки некоторых статей на эту тему.

Без оформления платной подписки единовременно можно создать только один список вещей. Если оформить подписку, кроме создания неограниченного списка вещей, появятся такие возможности, как копирование и восстановление списков, экспорт списков, сохранение резервных копий на Google Drive и др.



а) б) в)

Рисунок 1.3 – Мобильно приложение «PackKing»: а) ввод основных данных о списке, б) продолжение ввода данных, в) просмотр списка по категорям

Достоинства:

1) обширный набор характеристик, по которым составляется список;
2) наличие фильтров и ручной сортировки (перемещений) пунктов;
3) наличие различных способов просмотра списка: по категориям или по темам;
4) создание резервных копий списков;
5) любой пункт из основного списка можно добавить в отдельный список, называемый «список покупок», что помогает пользователю минимизировать время подготовки к поездке.

Недостатки:

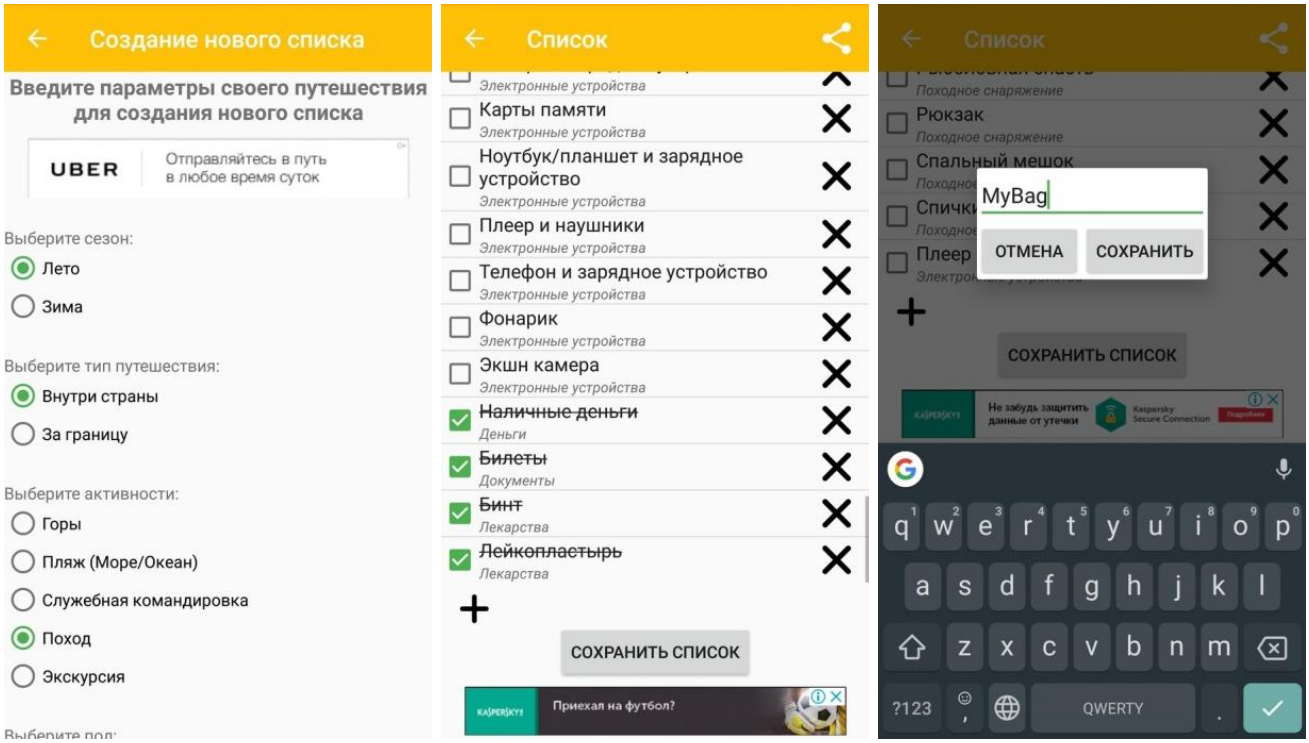
1) невозможно создание списка "с нуля" (необходимость выбора хотя бы одной темы);
2) для создания более одного списка требуется оформление платной подписки;
3) для оформления собственного списка из шаблонных необходимо не только отметить интересующие пункты, но и удалить ненужные, каждый пункт в отдельности, иначе бесполезные пункты будут занимать место на дисплее, тем самым затрудняя просмотр.

1.4 «Багаж – список вещей»

Мобильное приложение для Android, представленное на рисунке 1.4 [1]. Принцип работы: для создания нового списка необходимо указать сезон (лето или

зима), тип путешествия (имеется в виду поездка за границу или внутри страны), активность и пол путешественника.

По введенным данным будет сформирован предлагаемый приложением список вещей. Из данного списка необходимо удалить все ненужные пункты каждый по отдельности, иначе они буду включены в итоговый список. Можно создать собственные пункты. Как и в других приложениях, поддерживается возможность выделения пункта, при этом его текст зачеркивается, а сам пункт опускается в низ списка. Реализована возможность поделиться списком через Bluetooth, социальные сети, электронную почту и сообщения, копировать список, экспортировать в pdf-формат, сохранить на Google Drive и др.



а) б) в)

Рисунок 1.4 – Мобильное приложение «Багаж – список вещей»: а) ввод основных данных о списке, б) выбор пунктов списка, в) выбор названия списка и сохранение

Достоинства:

1) приложение простое и интуитивно понятное в использовании;
2) обширные шаблонные списки.

Недостатки:

1) ограниченное количество типов активностей (всего пять), можно выбрать только одну, невозможно создать пользовательский тип активности;
2) много обычной и полноэкранной рекламы;
3) возможность отключения рекламы является платной;
4) при добавлении собственного пункта обязательно указание категории;
5) категорий недостаточно, невозможно создать пользовательскую категорию;

6) некоторые категории являются узкоспециализированными (например, категория «деньги»);
7) под каждым пунктом прописана его категория, что ухудшает восприятие списка;
8) необходимость удаления всех ненужных пунктов, иначе все они сохранятся в итоговый список.

1.5 Выбор инструментов разработки

1.5.1 Java

Java – сильно типизированный объектно-ориентированный язык программирования [2]. Выпущен 23 мая 1995 года. Создателем языка Java является крупная американская компания по производству программного и аппаратного обеспечения Sun Microsystems (в 2009-2010 гг. продана корпорации Oracle).

В Java имеется только восемь примитивных типов данных, диапазоны которых жестко фиксированы для обеспечения платформенной независимости. Объекты создаются только динамически. В отличие от языков C, C++, в Java нет явных указателей. В данном языке существуют ссылки. Ссылка – это переменная объектного типа (объектный тип – это любой тип, кроме примитивного). Для доступа к полям и/или методом объекта по ссылке не требуется особых операций разыменования. Использование ссылок является безопасным инструментом программирования благодаря наличию фиксированных ограничений для их применения [3, 4].

В языке Java запрещено удаление объекта явным образом. Для этого реализован механизм сборки мусора. В этом случае, некий процесс, называемый сборщиком мусора, периодически удаляет из памяти неиспользуемые программой объекты. Для этого необходимо, чтобы на объект не указывала никакая ссылка. Поэтому программисту необходимо присвоить ссылке на ненужный объект значение null. При этом стоит учитывать, что данный объект не будет удален немедленно, но гарантированно, что это произойдет в ближайшее время.

В данном языке функция не может существовать вне класса. Поэтому в Java используется термин «метод», и все стандартные функции переведены в методы соответствующих классов [5].

Текст программы, написанной на Java, транслируется в байт-код, который, в свою очередь, обрабатывается JVM (Java Virtual Machine) – виртуальная Java машина. Байт-код – это машинно-независимый код низкого уровня, который будет интерпретироваться и компилироваться в машинный язык быстрее, нежели метод прямой интерпретации и компиляции исходного текста программы в машинный код. Байт-код может выполняться на любом оборудовании и на любой операционной системе при наличии соответствующей виртуальной машины, которая сможет выполнить данный байт-код. Использование виртуальной машины повышает безопасность выполнения программы. Любые действия программы, превышающие дозволенные ей операции, буду немедленно блокированы [6].

Одно из направлений использования языка Java – разработка мобильных приложений для операционной системы Android. Это самый популярный язык среди Android-разработчиков. Более того, исходный код Android написан именно на этом языке, существует подробная документация, а также множество обучающих курсов разработки мобильных приложений, написанных на Java. При этом исходный текст приложений транслируется в нестандартный байт-код, который будет исполняться на устройствах с операционной системой Android разработанной Google виртуальной машиной Dalvik, а с версии Android 5.0 виртуальную машину сменила Android Runtime (ART) – специальная среда выполнения для Android-приложений [7]. Dalvik применяет JIT-компиляцию, а ART компилирует текст программы при установке приложения, что увеличивает скорость выполнения приложения, а также экономит потребление приложением заряда батареи. Обратной стороной применения ART становится более длительная установка приложения, увеличивается занимаемое место приложением в памяти, а также это сказывается на времени включения самого устройства, на котором установлено приложение. ART использует тот же байт-код, что и Dalvik, для обеспечения обратной совместимости.

Кроме того, для компиляции Android-приложений необходимо установить Android SDK (Software Development Kit) – набор инструментов разработки для мобильных приложений под операционную систему Android, разработанный компанией Google. Он предоставляет функционал для отладки и тестирования приложения, что включается в себя определение совместимости приложения на на различных версиях Android, тестирование продукта на разных устройствах (например, мобильные телефоны, планшеты, наручные часы, телевизоры и другие гаждеты).

Основные поддерживаемые среды разработки.

1. Android Studio – свободная интегрированная среда для разработки приложений под Android. (с конца 2014 года является официальной средой для разработки мобильных приложений). Основана на программном обеспечении IntelliJ IDEA. Разработчик: Google. Поддерживаемые языки программирования: Java, C++, Kotlin.
2. NetBeans – свободная интегрированная среда для разработки программного обеспечения, написанного на языках Java, PHP, Python, JavaScript, C/C++ и др. Разработчики: NetBeans Community и NetBeans Org. Проект спонсируется Oracle.
3. Eclipse – свободная интегрированная среда разработки для написания приложений на различных языках. Основным направления является язык Java.Изначально создавалась IBM как IDE для написания программ на различных языках под платформы IBM. Нынешний разработчик: Eclipse Foundation.
4. IntelliJ IDEA – интегрированная среда разработки для различных языков, в том числе Java, Python, Kotlin, Ruby, JavaScript. Имеется как свободная

версия продукта (Community Edition), так и коммерческая (Ultimate Edition). Разработчик: компания JetBrains.

Достоинства.

1. Независимость от оборудования и операционной системы. Программы, написанные на Java, транслируются в байт-код, который выполняется специальной виртуальной машиной. Таким образом, такие программы могут выполняться на любом устройстве, для которого существует подобная виртуальная машина.
2. Виртуальная машина Java полностью контролирует выполнение программы, тем самым обеспечивая высокую степень безопасности.
3. Богатая библиотека. На протяжении своей истории язык накопил в себе большое количество стандартных алгоритмов и типов данный, что во многом облегчает работу разработчика.
4. Запрет на прямую работу с памятью. Отсутствие в языке таких не устойчивых средств работы с памятью, как указатели в языках C/C++.
5. Сборка мусора.
6. Оптимизация байт-кода с помощью JIT-компилятора.

Недостатки.

1. Обратная сторона богатой библиотеки заключается в том, что для новичка язык является сложным, и изучение его до некоторого уровня требует значительных временных ресурсов.
2. Снижение производительности из-за наличия средств для повышения степени надежности.

1.5.2 C#

C# - объектно-ориентированный язык программирования [8]. Создан группой разработчиков компании Microsoft в 2000 году. Изначально разрабатывался как язык для написания программ под платформу Microsoft. NET Framework.

В языке C# реализована сборка мусора, подобная сборке мусора в Java, которая направлена на поиск и удаление из памяти более не используемых программой структур. Внедрен продуманный и расширяемый механизм надежной обработки исключений, способствующий выявлению ошибок и дальшейней работы с ними. Язык обладает строгой типизацией, ограждающей выполнение несогласованного приведения типов, обращение к неинициализированным переменным, а также выход за пределы массива [9]. Все эти факторы повышают устойчивость и надежность программ, написанных на языке C#.

Все типы C#, в том числе и примитивные, являются наследниками общего типа object. Вследствие этого, все типы имеют общий набор операций, а знания разных типов могут обрабатываться схожим способом. Кроме того, в C# возможно создавать пользовательские типы значений и ссылочные типы, что позволяет использовать как динамическое выделение памяти, так и хранение значений в стеке [10].

Разработка мобильных приложений не является его главным направлением, и здесь больше ориентирован на создание игр. Созданный намного позже появления Java язык C# является более современным и неперегруженным языком, что, несомненно, привлекает внимание, особенно новичков. Однако C# не предоставляет широкие возможности для Android-разработчиков.

Основные поддерживаемые среды разработки.

1. Visual Studio - набор продуктов компании Microsoft, содержащий интегрированную среду разработки программного обеспечения и некоторые другие инструменты. Данное программное решение позволяет разрабатывать и консольные приложения, и приложения с графическим интерфейсом, а также веб-приложения, веб-сайты. Visual Studio содержит редактор исходного кода с технологией IntelliSense (технология автодополнения от компании Microsoft) и с функционалом обычного рефакторинга кода. Встроенный отладчик работает на двух уровнях: машинном и исходном. Прочие инструменты разработки: редактор форм, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio имеет возможность создавать новые и подключать сторонние дополнения для увеличения возможностей на различных уровнях, в том числе включение поддержки системы контроля версий исходного кода.
2. Xamarin Studio – интегрированная среда разработки для создания мобильных приложений для iOS, Android, Windows Phone. Данная среда предназначена для операционных систем Windows и Mac OS X. Имеет схожую с Visual Studio функциональность: рефакторинг, подсветка синтаксиса, автодополнение кода, поиск и навигация, отладка, интеграция с системами контроля версий.
3. MonoDevelop – Свободная интегрированная среда разработки. Поддерживаемые языки программирования: C#, Java, C/C++, Boo, Nemerle. Разработчик: MonoDevelop Team. Основная функциональность: навигация по коду, возможность рефакторинга, стандартные шаблоны проектов, автоматическое дополнение кода, работа с базами данных, поддержка создания Unit-тестов, автоматическое создание документации, интеграция с Microsoft Visual Studio, расширение функциональности посредством включения плагинов и внешних инструментов, управление исходным кодом (Subvertion).

Достоинства.

1. Язык легкий в изучении (не труднее Java).
2. Строгая типизация повышает надежность.
3. Сборка мусора.
4. Скорость выполнения программ, написанных под ОС Android, выше, чем у программ, написанных на Java.

Недостатки.

1. Глубокая ориентированность на Windows платформу. C# может использоваться для разработки программного обеспечения под другие платформы, но зачастую возникают проблемы, связанные с отсутствием Windows окружения. Как следствие - низкая степень кросплатформенности и дополнительные расходы на ее повышение.
2. Объем библиотеки языка C# невелик по сравнению с библиотеками других популярных языков программирования.

1.5.3 React Native

React Native – недавно появившийся на рынке инструмент для создания нативных мобильных приложений под операционные системы iOS и Android. Нативными называются приложения, написанные для конкретной операционной среды и для определенных устройств. Тем самым они максимально используют возможности операционной среды, для которой были написаны, а также получают доступ к аппаратной части устройств, например, могут использовать камеру мобильного устройства, микрофон, геолокацию, адресную книгу и т.п. Кроме того, такие приложения работают быстрее и надежнее [11].

React Native содержит JS-библиотеку React, предназначенную для написания пользовательских интерфейсов. Данный продукт будет особенно удобен для веб-разработчиков, которые смогут, не покидая привычной для себя среды и языка JavaScript, написать мобильное приложение. В ходе работы над мобильным приложением понадобятся классические инструменты разработки, а также программное обеспечение для работы с JavaScript, такие как текстовый редактор и отладчик [12].

Достоинства.

1. Использование одних и тех же инструментов для написания программных продуктов для разных платформ.
2. Использует быстрый и популярный язык программирования JavaScript
3. Благодаря использованию виртуального DOM (Document Object Model) увеличивается скорость разработки и улучшается опыт пользователя.
4. Использование единой кодовой базы для создания приложения под Android и iOS не влияет на производительность выполнения программ.

Недостатки.

1. Продукт находится в активной стадии разработки (возникают проблемы, такие как неполные библиотеки).
2. Высокие темпы развития. Продукт часто обновляется и совершенствуется, что требует от разработчиков постоянного изучения новых приемом программирования.
3. Непроработанная и неполная документация вследствие высоких темпов развития программного обеспечения.

1.6 Вывод по разделу

После рассмотрения существующих конкурентов стало понятно, что разрабатываемое приложение сможет занять достойное место на рынке и найти своих пользователей, за счет своих преимуществ перед другими подобными приложениями.

В результате анализа существующих инструментов разработки для платформы Android было принято решение вести разработку приложения на языке Java, способном обеспечить достаточную скорость и гибкость разработки.

2 МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ СБОРА ЛИЧНЫХ ВЕЩЕЙ В ДОРОГУ

Для проектирования и визуализации структуры программного продукта и связей между его компонентами на различных уровнях был использован унифицированный язык моделирования UML (Unified Modeling Language) [13].

2.1 Логика работы программы

Для визуализации общего функционала приложения и описания акторов была построена диаграмма вариантов использования, представленная на рисунке 2.1.

Приложение рассчитано на один тип действующих лиц.

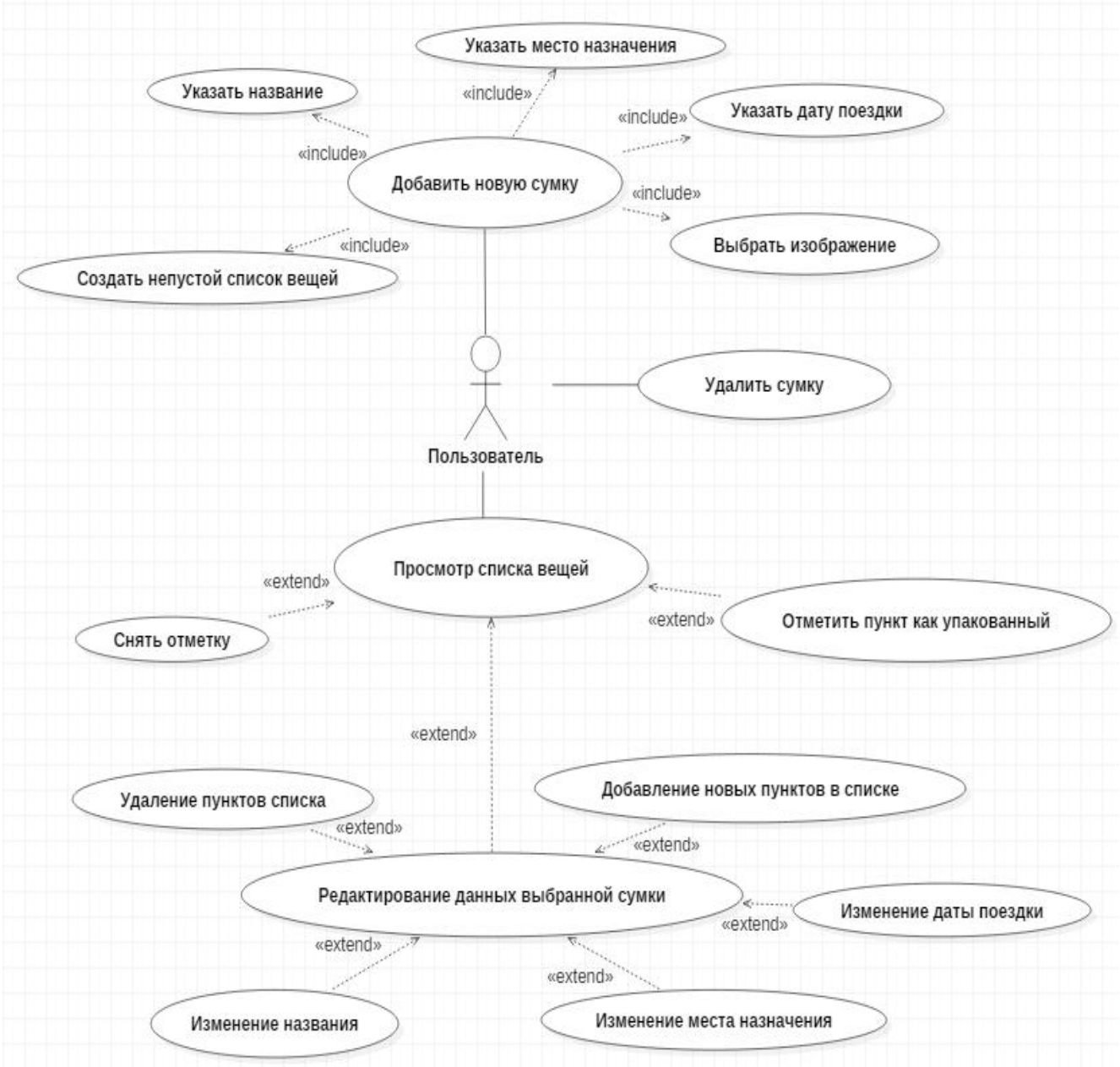


Рисунок 2.1 – Диаграмма вариантов использования

Основными действиями пользователя при использовании данного мобильного приложения являются формирование новых списков вещей, их просмотр и удаление.

Создание новой сумки включается в себя несколько действия, а именно, указание названия формируемой сумки, места назначения, даты поездки/мероприятия, выбор схематичного изображения списка, которое будет использоваться вместе с названием на начальном экране для идентификации списка, и создание самого списка вещей посредством последовательного ввода всех его пунктов.

Действие «просмотр списка вещей» может быть расширено такими действиями, как установка/снятие отметок около пунктов в списке, редактирование списка, где возможны следующие варианты использования: добавление/удаление пунктов списка, изменение названия сумки, места назначения, даты поездки/мероприятия.

2.2 Архитектура программного продукта

Для демонстрации классов программы, их свойств и методов, а также связей между ними была построена диаграмма классов, представленная на рисунке 2.2.

Activity представляет собой отдельное окно приложения. Для каждой activity создан собственный класс, который является потомком класса AppCompatActivity. AppCompatActivity – это базовый класс для activity с поддержкой панели инструментов в верхней части мобильного приложения.

Классы-адаптеры созданы для корректного отображения данных в соответствующих графических компонентах экрана. Данные классы с классами activity образуют отношения композиции [14, 15].

Activity-классы взаимодействую друг с другом, обмениваясь данными, как показано на схеме.

2.3 Структура проекта

Диаграмма компонентов является физическим описанием системы. В данном случае она отражает файловую структуру проекта (рисунок 2.3).

Данное мобильное приложение, как и другие Android-приложения, после компиляции упаковано в один исполняемый apk-файл, который содержит в себе все файлы проекта, такие как файл манифеста (AndroidManifest.xml), dex-файл, файл ресурсов.

Файл манифеста содержит основную информацию о программе, которая необходима операционной системе Android. Данный файл описывает компоненты приложения, что необходимо для вызова классов, которые реализуют эти компоненты. Также на основе описаний компонентов система Android узнает, при каких условиях их можно запускать. Кроме того, файл манифеста объявляет имя пакета приложения для его идентификации, в нем прописывается минимальный уровень API Android, необходимые для корректной работы приложения. Все подключенные библиотеки в приложении перечисляются в файле манифеста [16].

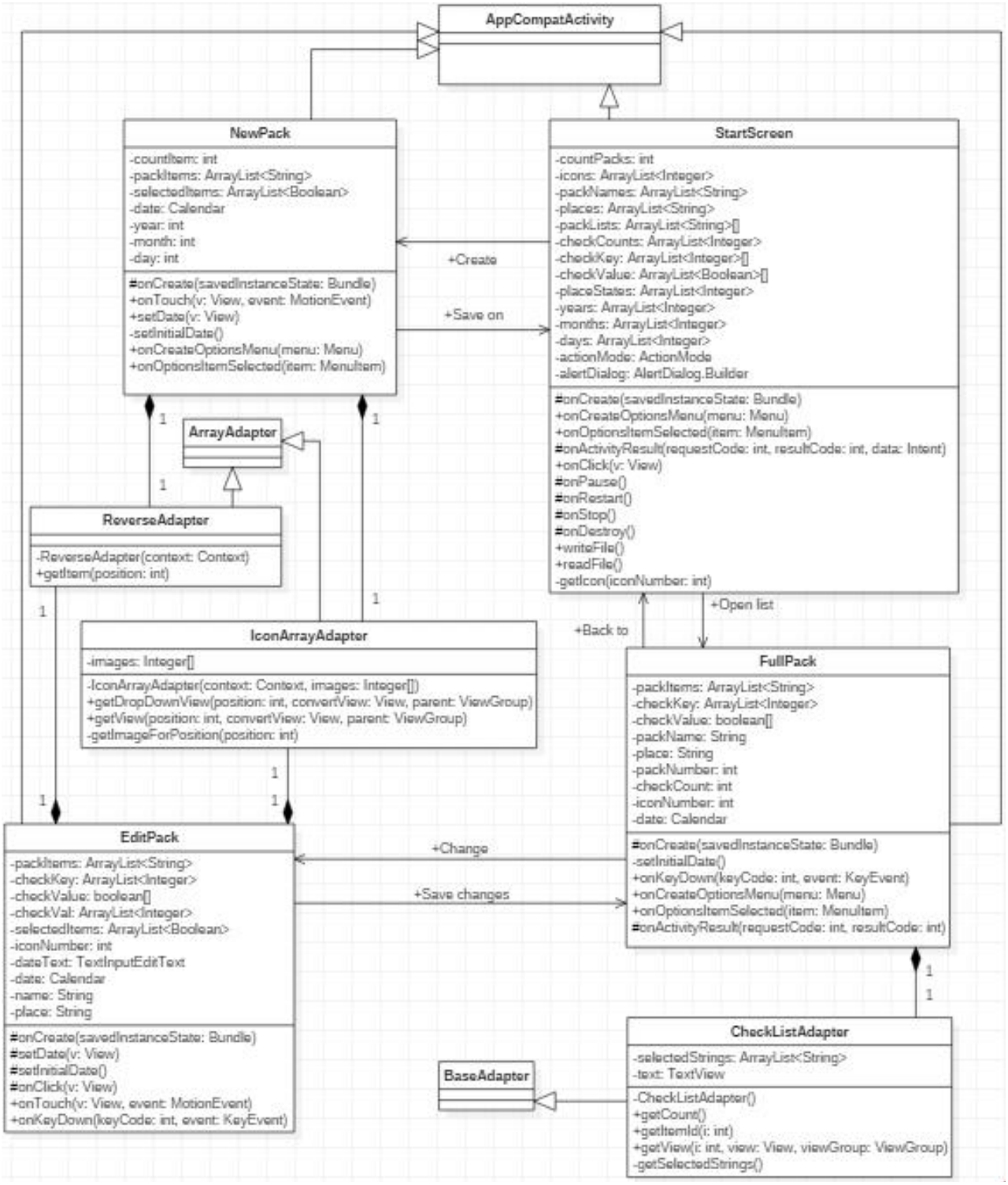


Рисунок 2.2 – Диаграмма классов

Dex-файл содержит скомпилированный код приложения. Файл ресурсов включает в себя xml-файлы, в которых находятся описания макетов всех окон приложения, панели инструментов и контекстного меню, цвета, строковые константы, векторные изображения иконок, стили и т.д.

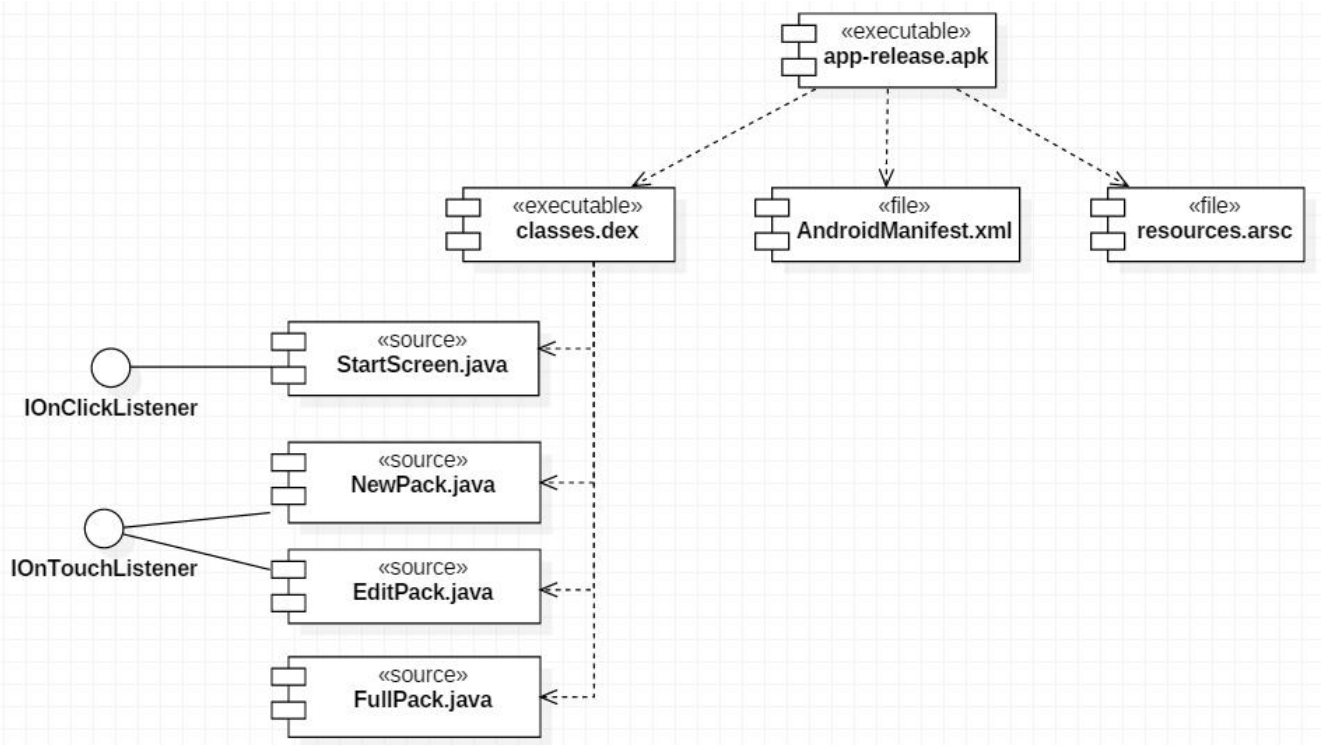


Рисунок 2.3 – Диаграмма компонентов

2.4 Вывод по разделу

В данном разделе была построена архитектура приложения, основанная на диаграммах вариантов использования, классов, компонентов.

3 АЛГОРИТМЫ СИСТЕМЫ

3.1 Общий алгоритм работы системы

На рисунке 3.1 представлен общий алгоритм работы системы. Во время запуска приложения начитается его инициализация, после чего система ожидает сообщения от пользователя для дальнейшей его обработки.

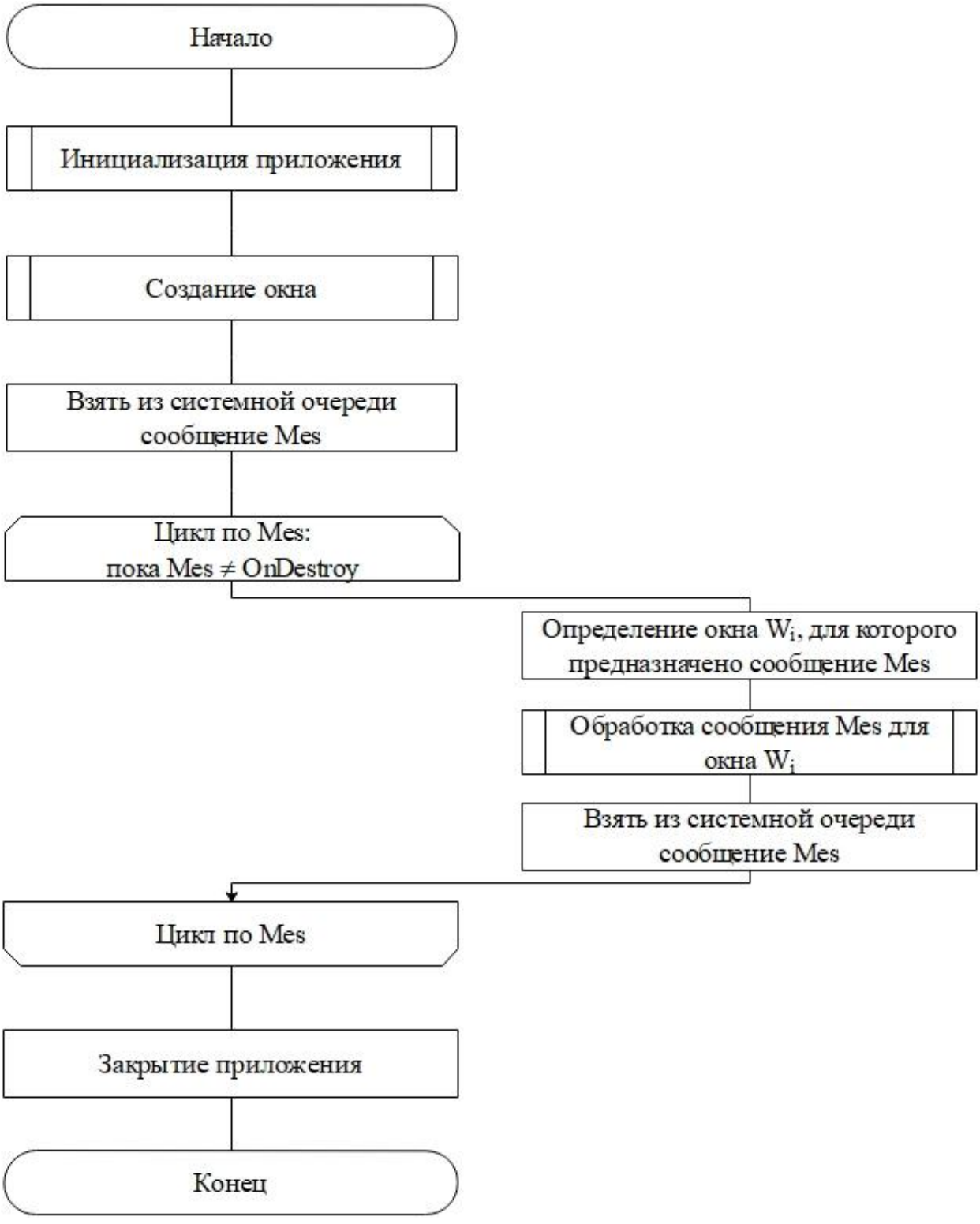


Рисунок 3.1 – Общий алгоритм работы системы

3.2 Алгоритм обработки сообщений пользователя

Алгоритм обработки сообщений пользователя из начального экрана приложения представлен на рисунке 3.2.

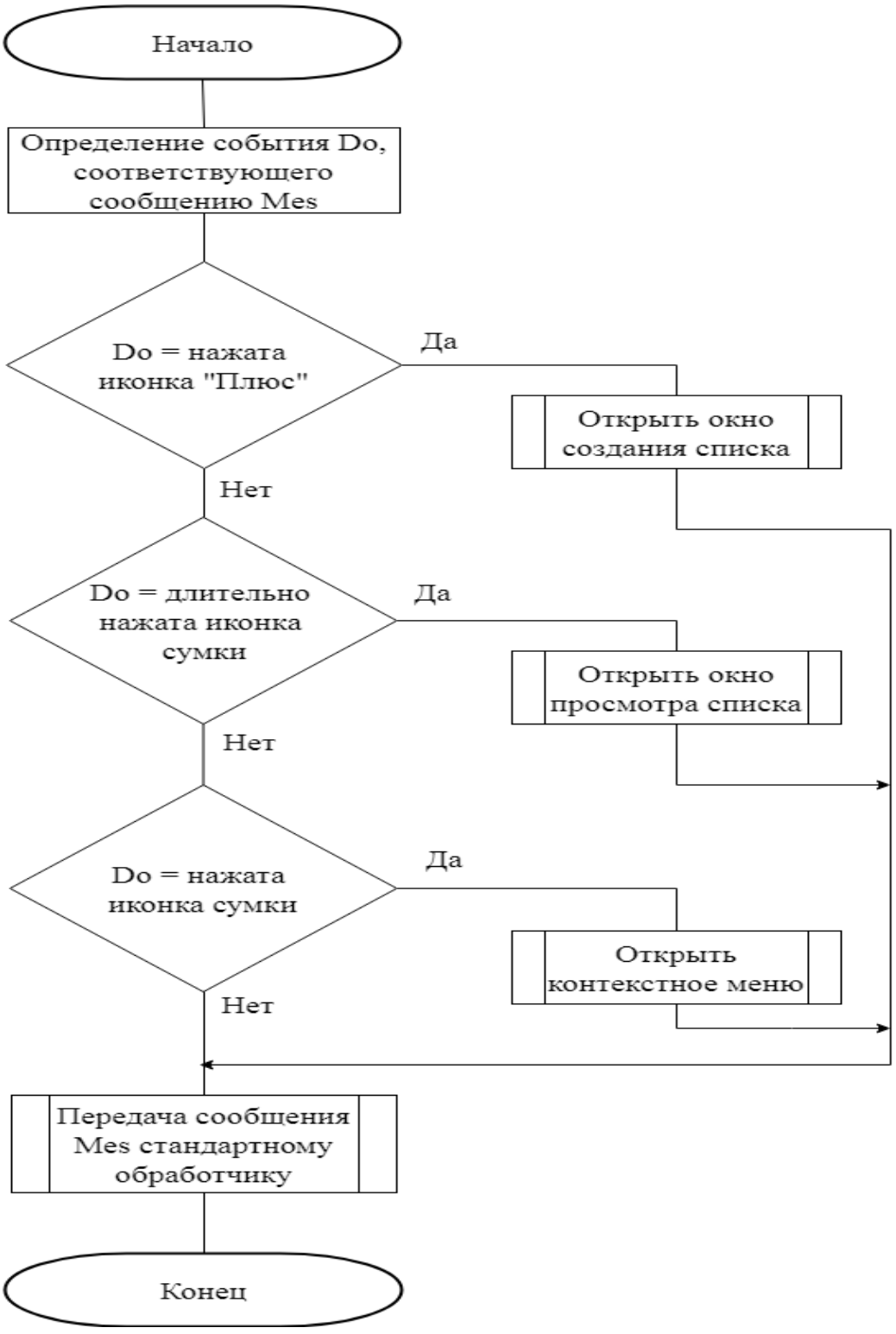


Рисунок 3.2 – Алгоритм обработки сообщений пользователя

Если пользователь нажал на иконку создания списка, системы открывает окно создания нового списка. При нажатии на иконку сумки система открывает окно

просмотра. Возможно длительное нажатие на иконку сумки пользователем. В этом случаем система создает соответствующее контекстное меню.

3.3 Алгоритм обработки вводимой пользователем информации

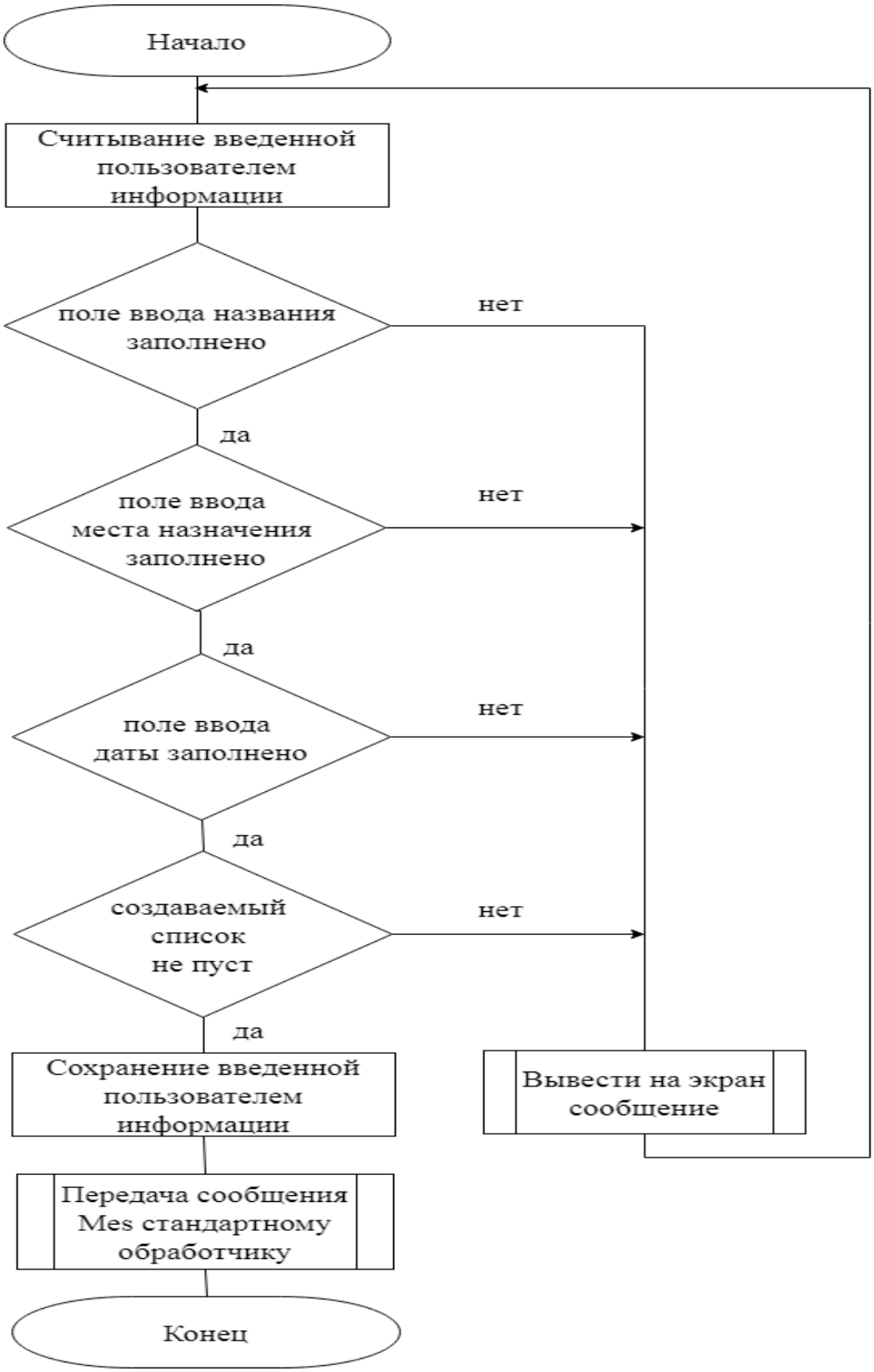


Рисунок 3.3 – Алгоритм обработки вводимой пользователем информации

Пользователь во время работы с приложением может вводить информацию в специальные редактируемые поля. На рисунке 3.3 представлен алгоритм обработки данного действия.

Программа поочередно проверяет, все ли поля заполнены. Если нет, будет выведено сообщение с указанием, какое поле необходимо заполнить. В противном случае, программа сохраняет введенную информацию.

3.4 Алгоритм обработки длительного нажатия на элемент окна

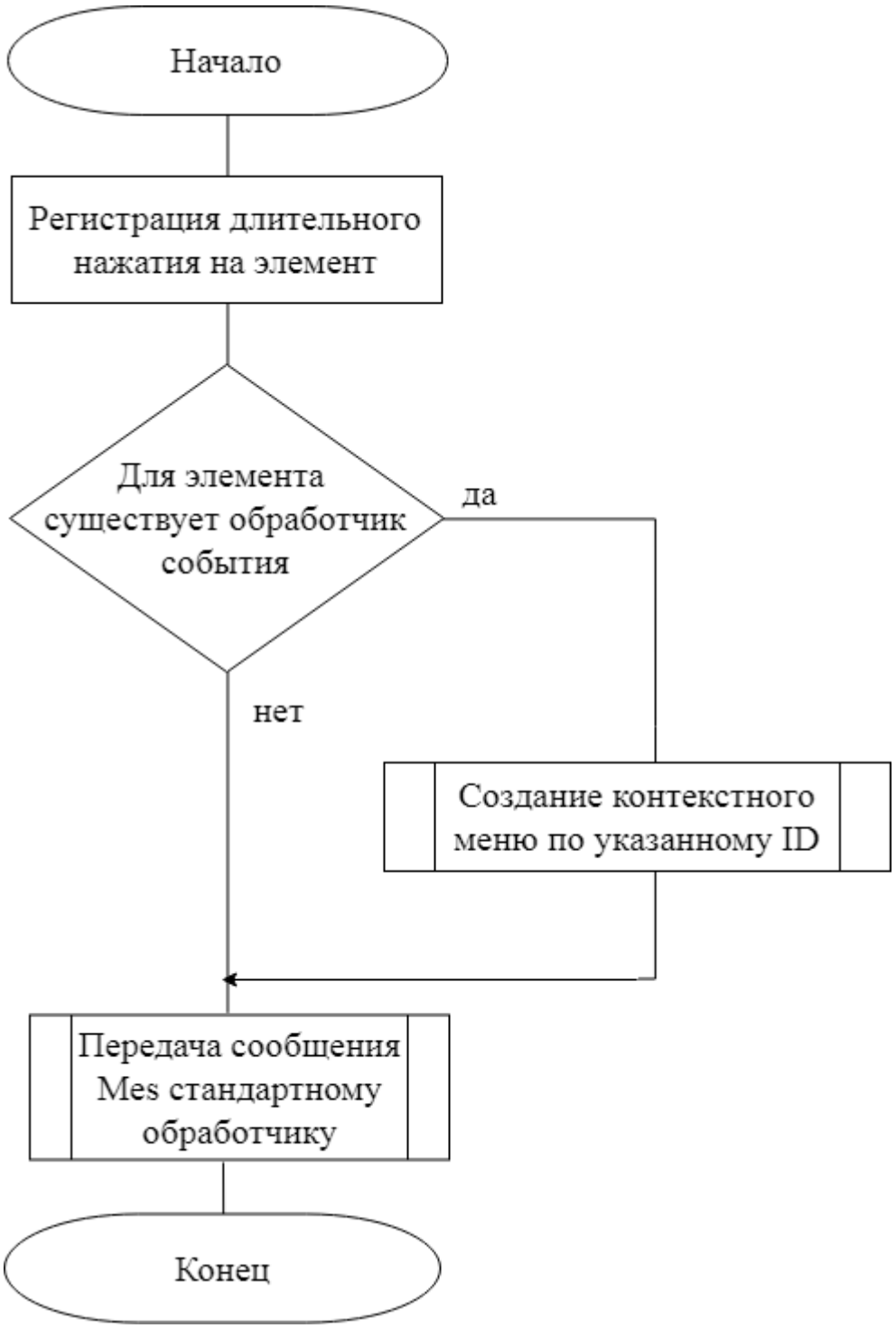


Рисунок 3.4 – Алгоритм обработки длительного нажатия на элемент окна

При длительном нажатии на элемент окна, для которого данный вид воздействия определен, приложение будет создавать соответствующее контекстное меню, данные о котором передаются системе по его идентификатору, как показано на рисунке 3.4.

3.5 Алгоритм обработки нажатий на пункты списка во время его просмотра

Функциональное требование о создании отметок пунктов списка реализует следующий алгоритм, представленный на рисунке 3.5.

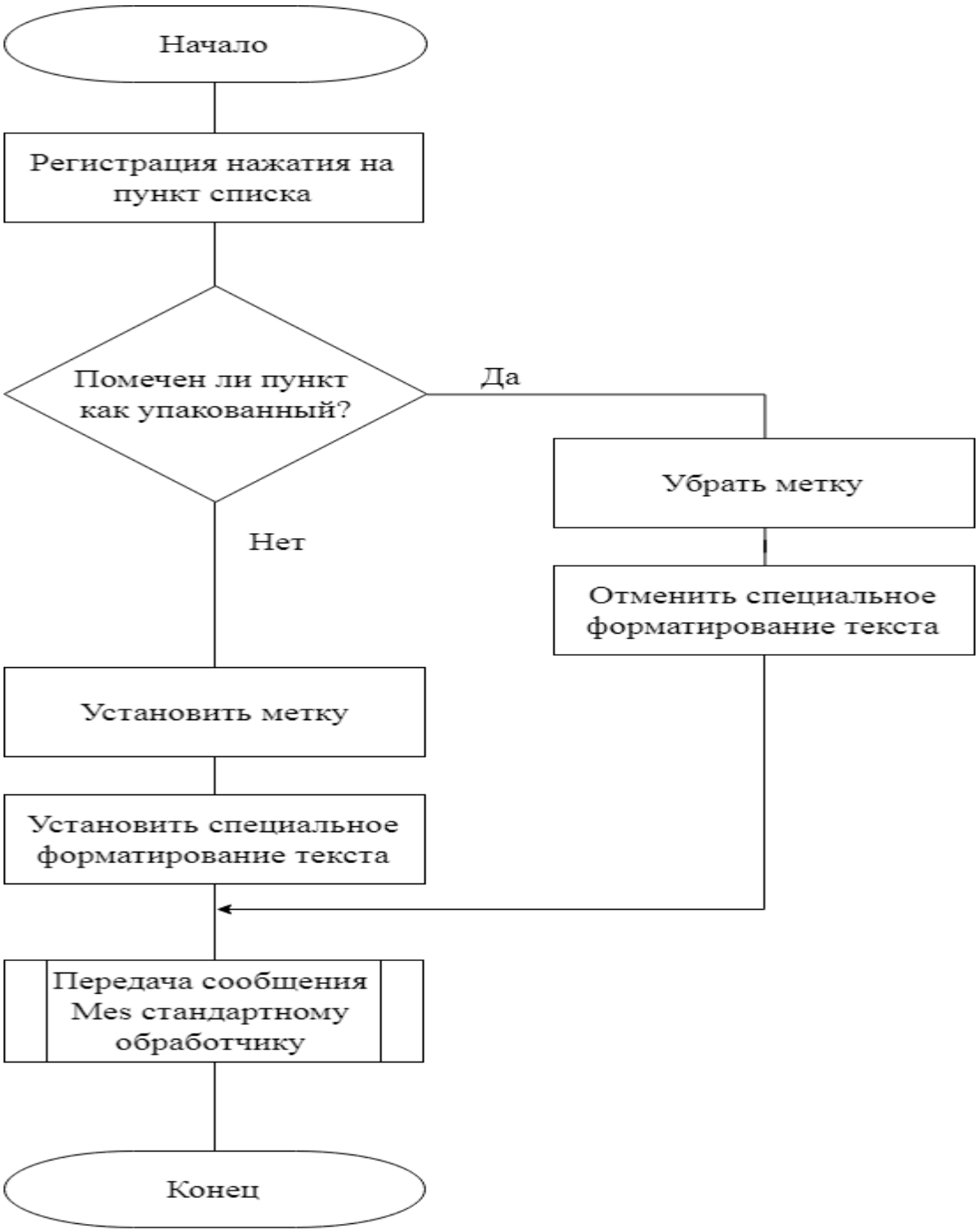


Рисунок 3.5 – Алгоритм обработки нажатий на пункты списка во время его просмотра

При нажатии на пункт списка, приложение проверяет, является ли данный пункт отмеченным. Если нет, выбранный пункт заносится в массив отмеченных пунктов, и для него применяется специальное форматирование. В противном случае, пункт удаляется из массива отмеченных пунктов, и для него отменяется специальное форматирование.

3.6 Вывод по разделу

В данном разделе были разработаны и показаны в виде блок-схем основные алгоритмы работы приложения, созданные для реализации поставленных ранее функциональных требований.

4 РАЗРАБОТКА ИНТЕРФЕЙСА И ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

4.1 Тестирование работоспособности приложения

В данном разделе приводится демонстрация интерфейса пользователя и одновременное тестирование приложения на конкретных задачах.

Сверху начального экрана приложения на панели инструментов расположены название приложения и иконка «Плюс» для создания нового списка вещей. Ниже находится текстовая подсказка для пользователя. Далее буду располагаться все созданные пользователем списки вещей в виде крупной иконки сумки и название списка под ней (рисунок 4.1).

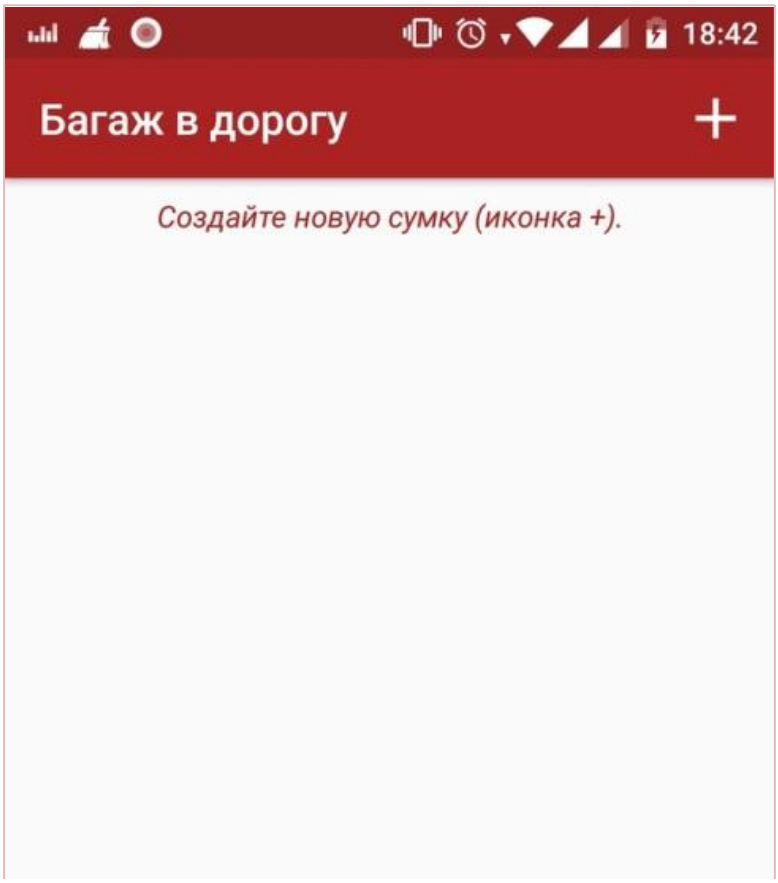


Рисунок 4.1 – Начальный экран

Для того чтобы создать новый список вещей, необходимо нажать на иконку «Плюс» в верхней части экрана. Появится новое окно, в котором нужно указать название нового списка вещей, место назначения, дату поездки, выбрать из выпадающего списка изображение сумки, которое будет находиться на начальном экране приложения, и далее ввести все необходимые пункты списка (рисунки 4.2 – 4.3). Для этого в поле с подсказкой «Что берем с собой?» нужно ввести название предмета и нажать кнопку Enter на клавиатуре. Новый пункт будет помещен в создаваемый список, расположенный ниже поля ввода (рисунок 4.4).

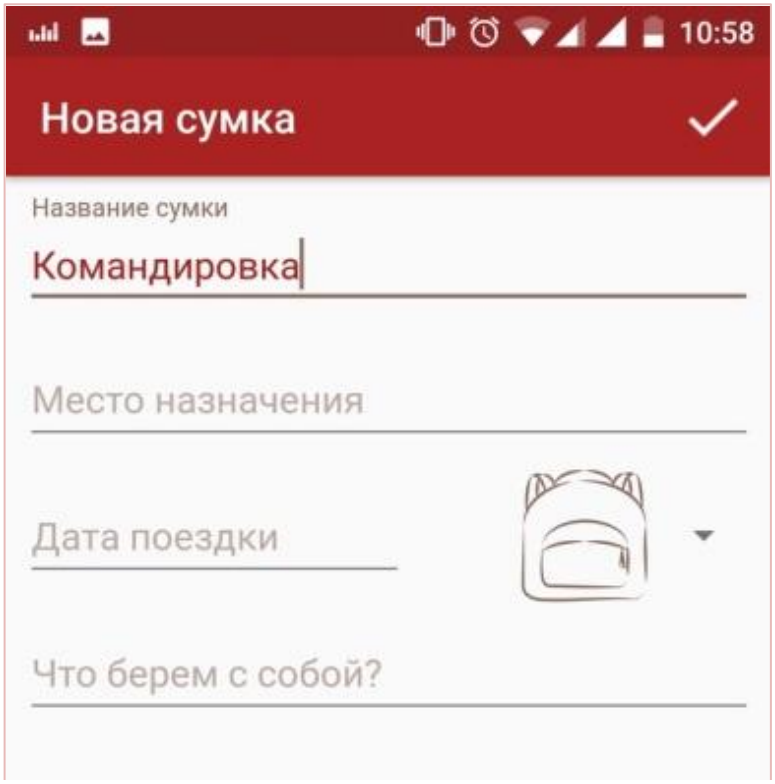
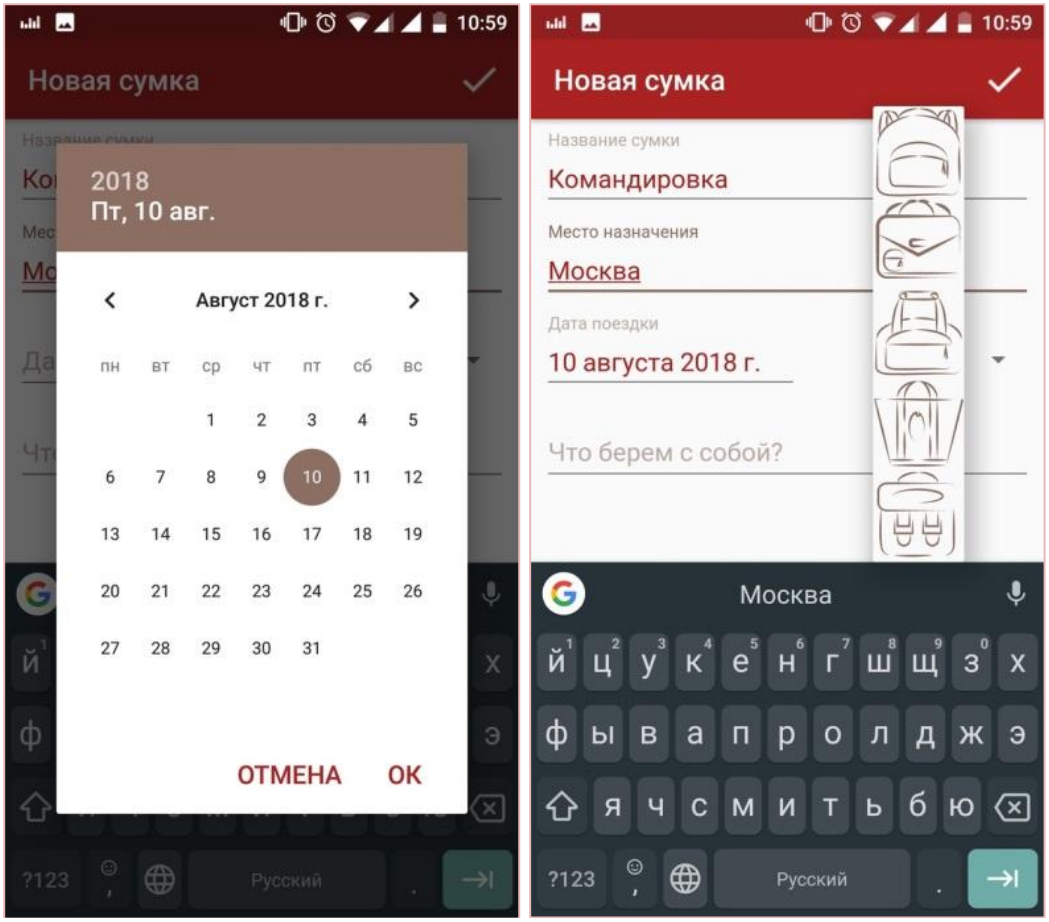
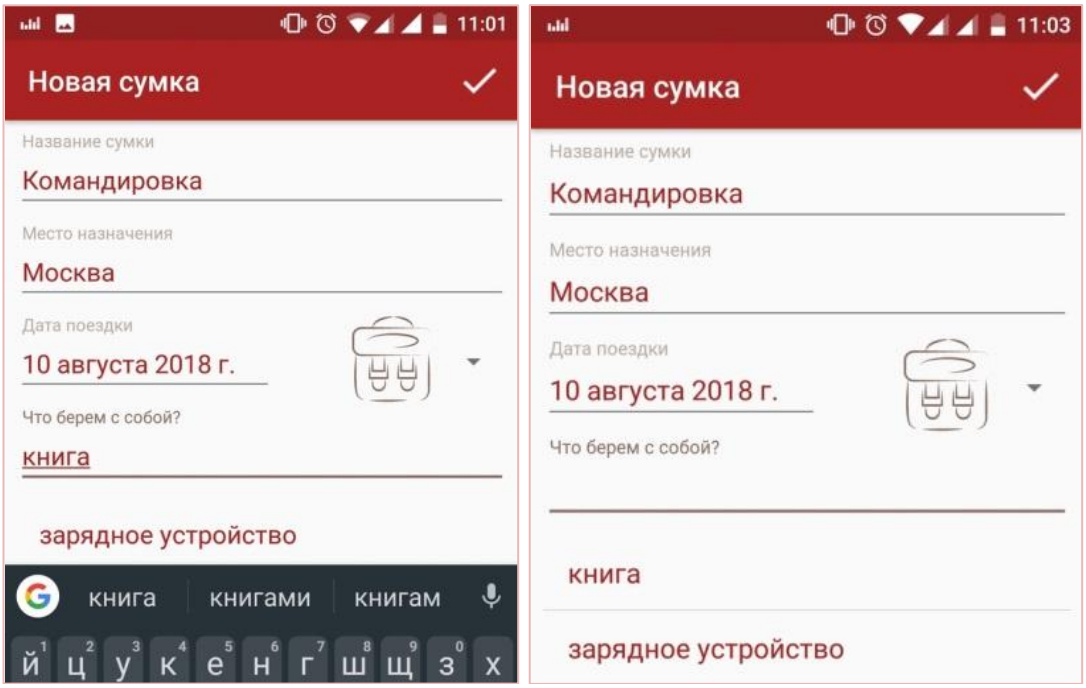


Рисунок 4.2 – Создание нового списка



а) б)

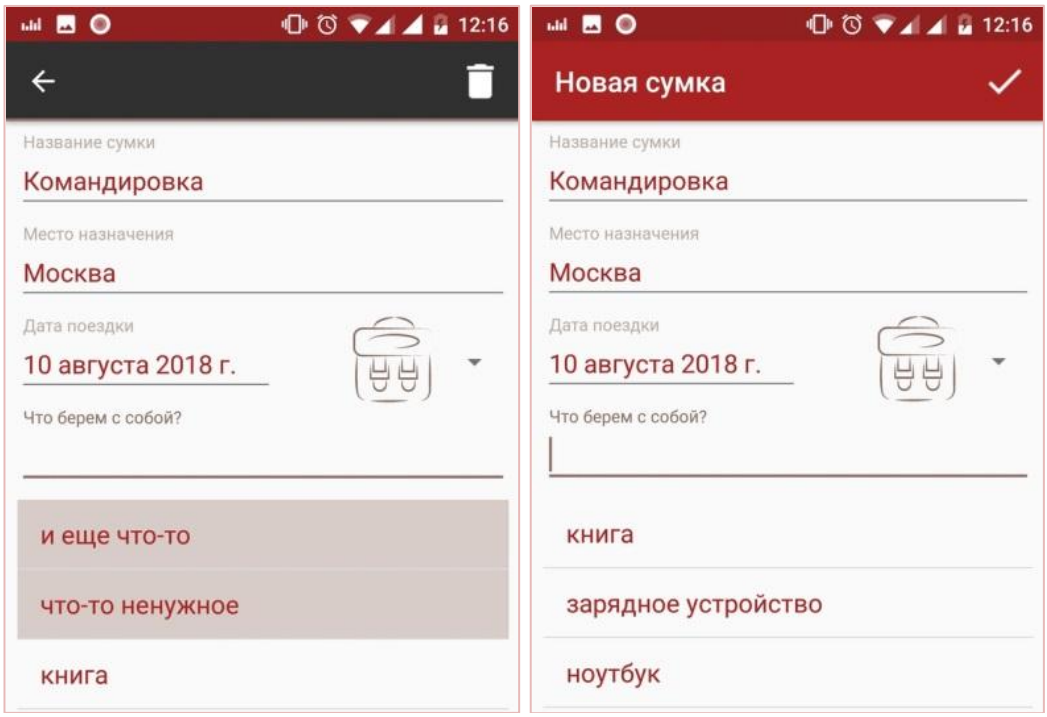
Рисунок 4.3 – Создание нового списка (1): а) выбор даты мероприятия, б) выбор изображения



а) б)

Рисунок 4.4 – Создание нового списка (2): а) ввод нового пункта списка, б) занесение нового пункта в список

Случайно занесенный неверный пункт в списке можно удалить, длительно нажав на него (рисунок 4.5).

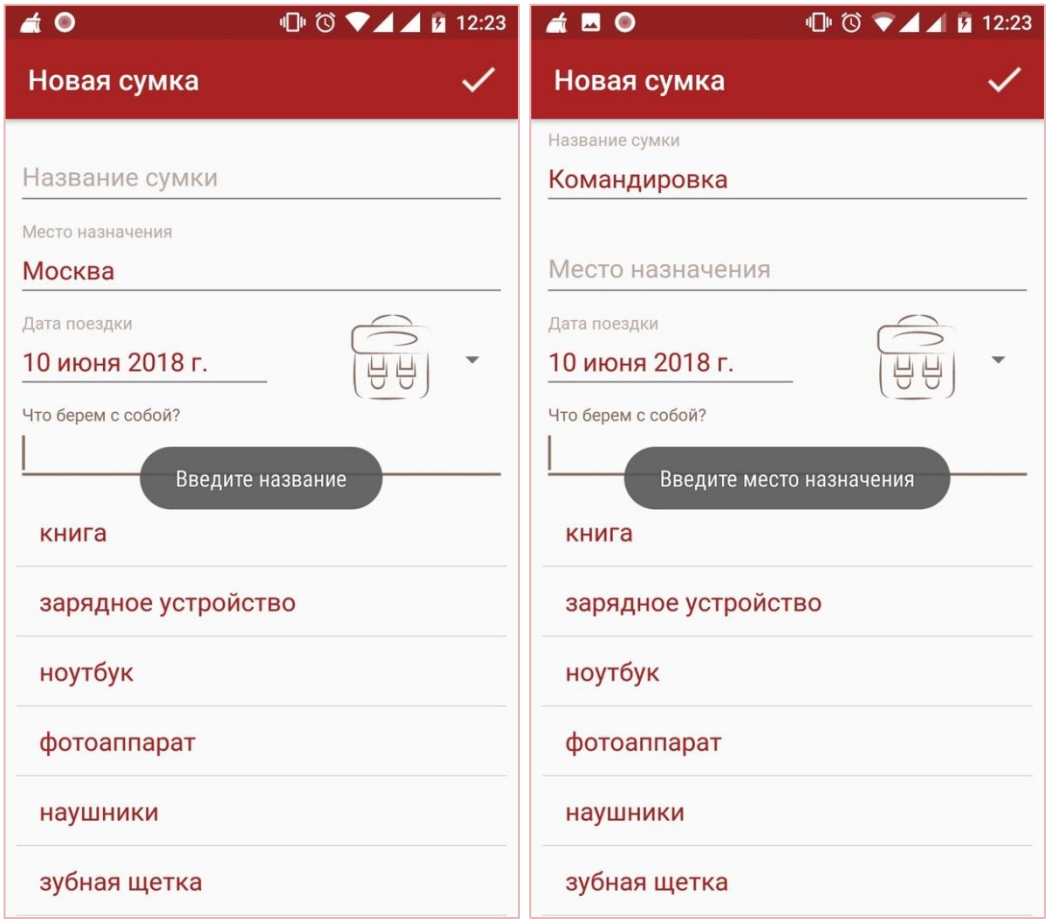


а) б)

Рисунок 4.5 – Создание нового списка (3): а) выделение пунктов для удаления, б) список после удаленя пунктов

За раз можно выделить несколько пунктов. При выделении пунктов появится контекстное меню с иконкой «Корзина». При нажатии на нее удалятся выделенные пункты.

При создании нового списка вещей все редактируемые поля должны быть заполнены. Иначе будет выведена текстовая подсказка с указанием, какое поле пользователь еще не заполнил. На рисунке 4.6 продемонстрирован случай не заполненых редактируемых полей «Название сумки» и «Место назначения». Для того чтобы сохранить создаваемый список необходимо заполнить данные поля.



а) б)

Рисунок 4.6 – Создание нового списка (4): а) не введено название сумки, б) не введено место назначения

На рисунке 4.7 продемонстрирован случай не заполненого редактируемого поля «Дата поездки». Для сохранения создаваемого списка необходимо заполнить данное поле через специальное диалоговое окно выбора даты. Для его вызова необходимо надать на область ввода текста в редактируемом поле.

Если пользователь попытается ввести строку текста, которая уже является пунктом списка, то при попытке занесения этой строки в спискок появится текстовая подсказка, как на рисунке 4.8.

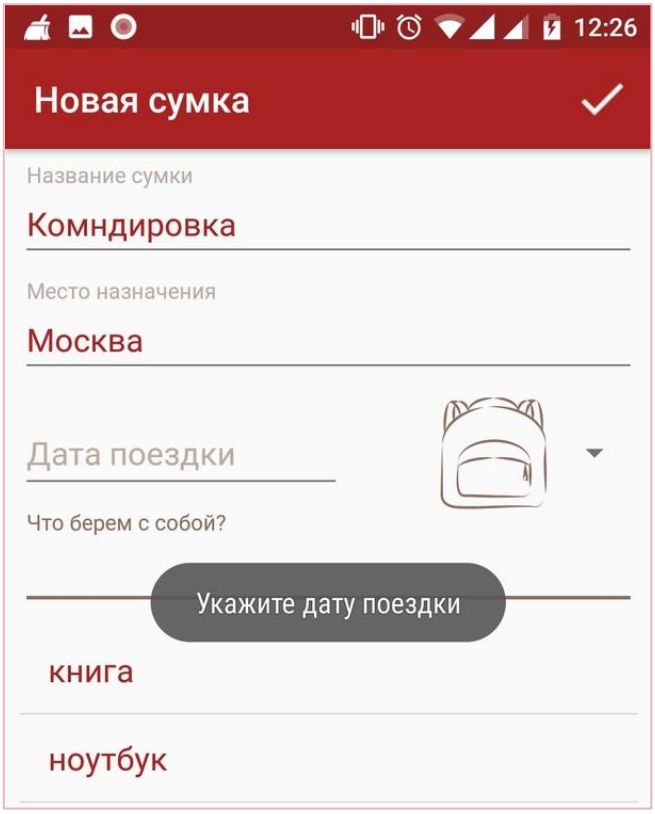


Рисунок 4.7 – Пустое поле ввода «Дата поездки»

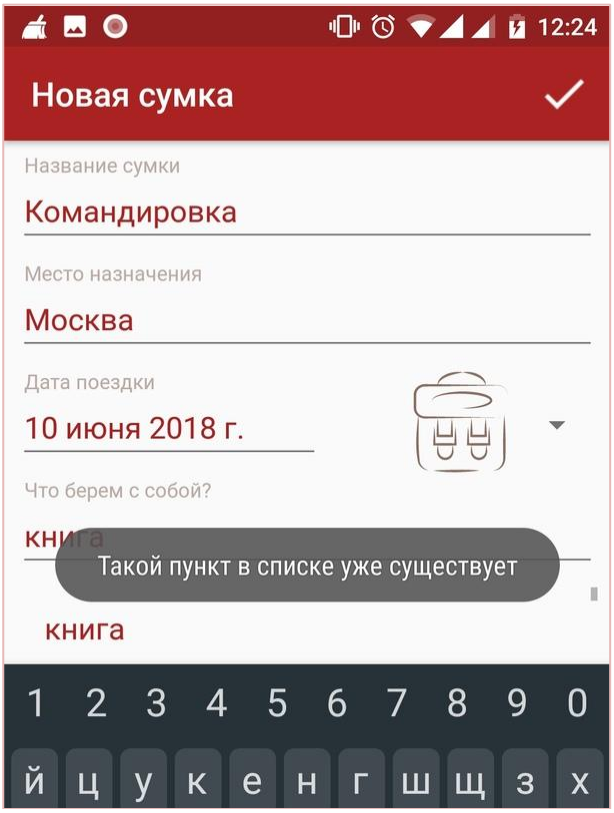


Рисунок 4.8 – Попытка ввода существующего пункта в списке

Также создаваемый список будет недоступен для сохранения, если в нем не будет указан хотя бы один пункт. Данная ситуация продемонстрирована на рисунке 4.9.

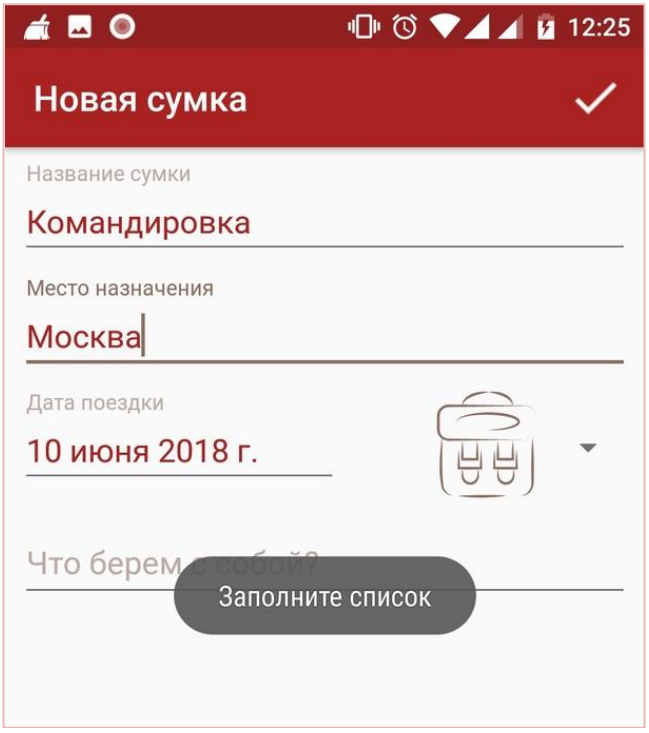
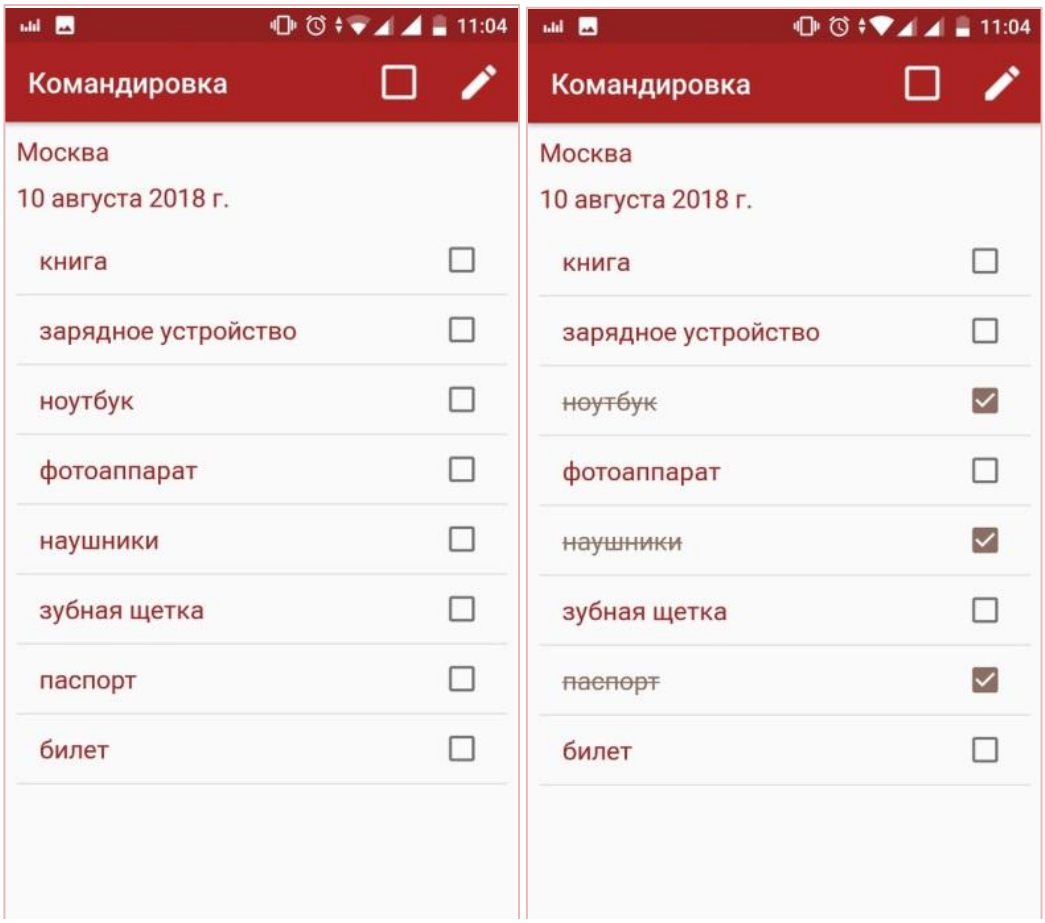


Рисунок 4.9 – Попытка сохранения пустого списка вещей

После ввода всех данных для сохранения необходимо нажать на иконку «Сохранить» (галочка) в верхней части экрана. После этого будет открыт начальный экран, на котором появится только что созданный список вещей (рисунок 4.10).

Рисунок 4.10 – Начальный экран с созданным списком

Для просмотра данных списка необходимо нажать на его изображение. В появившемся окне будет указана вся информация и сам список. Пункт списка можно выделить как упакованный, при этом напротив него появится галочка, а его текст станет зачеркнутым. Пункт можно восстановить повторным нажатием на него (рисунок 4.11).



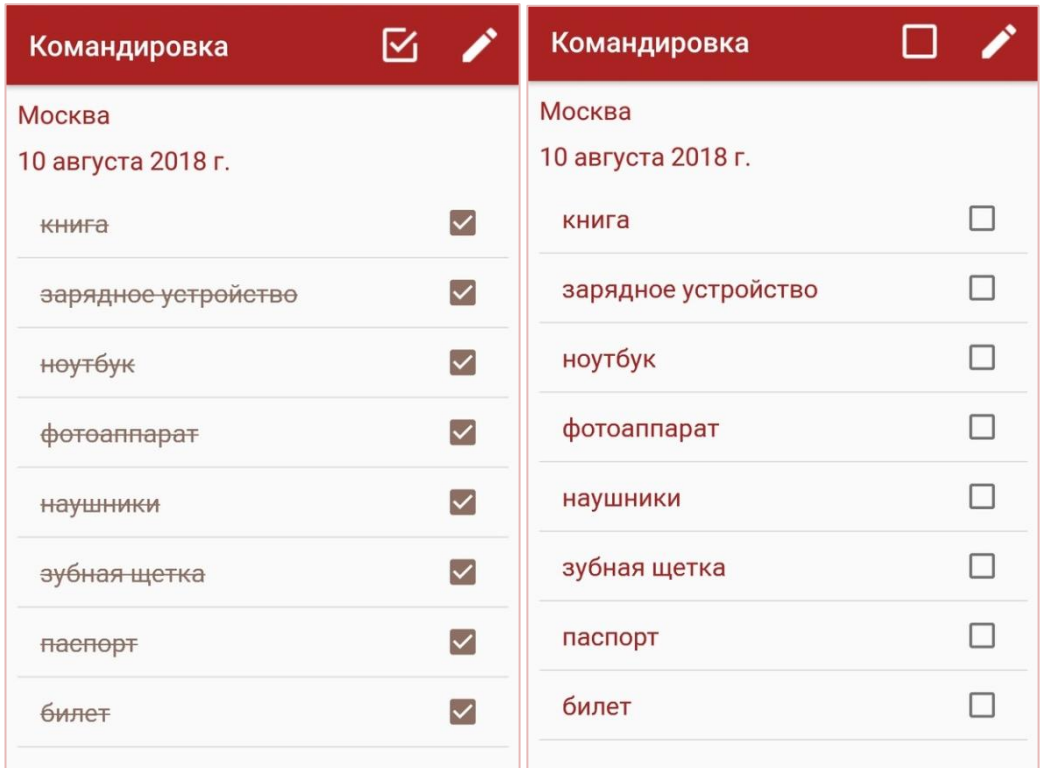
а) б)

Рисунок 4.11 – Просмотр списка: а) вид окна просмотра, б) выделение пунктов

Чтобы отметить единовременно все пункты, необходимо нажать на иконку пустого квадрата в верхней части экрана (рисунок 4.12). Повторное нажатие на данную иконку, в которой появляется галочка, снимает отметки со всех пунктов списка.

Для редактирования выбранного списка из окна просмотра необходимо нажать на иконку «Карандаш» в верхней части экрана. Откроется окно редактирования, в котором пользователь может изменить любые данные списка (рисунок 4.13).

Пользователь может изменить название списка, место назначения, дату поездки, схематичное изображение, добавить недостающие пункты и удалить ненужные.



а) б)

Рисунок 4.12 – Просмотр списка (1): а) все пункты отмечены, б) снятие всех отметок

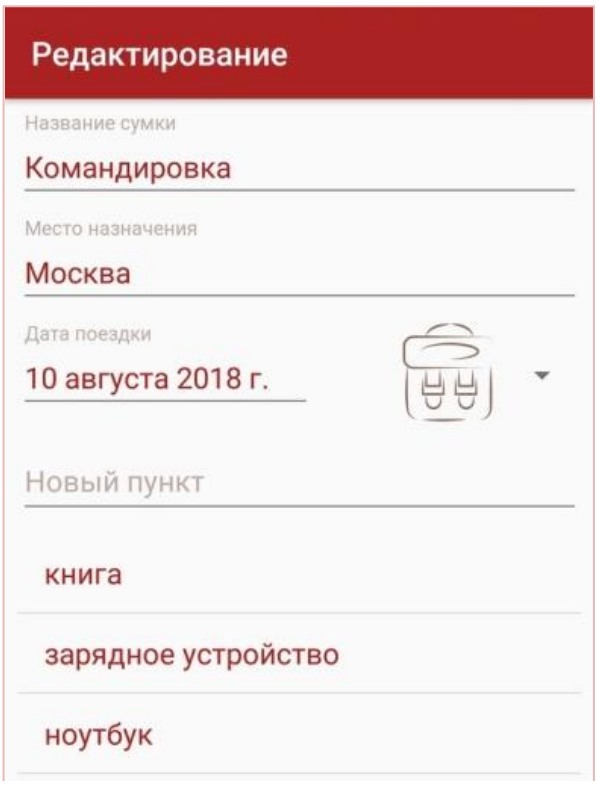
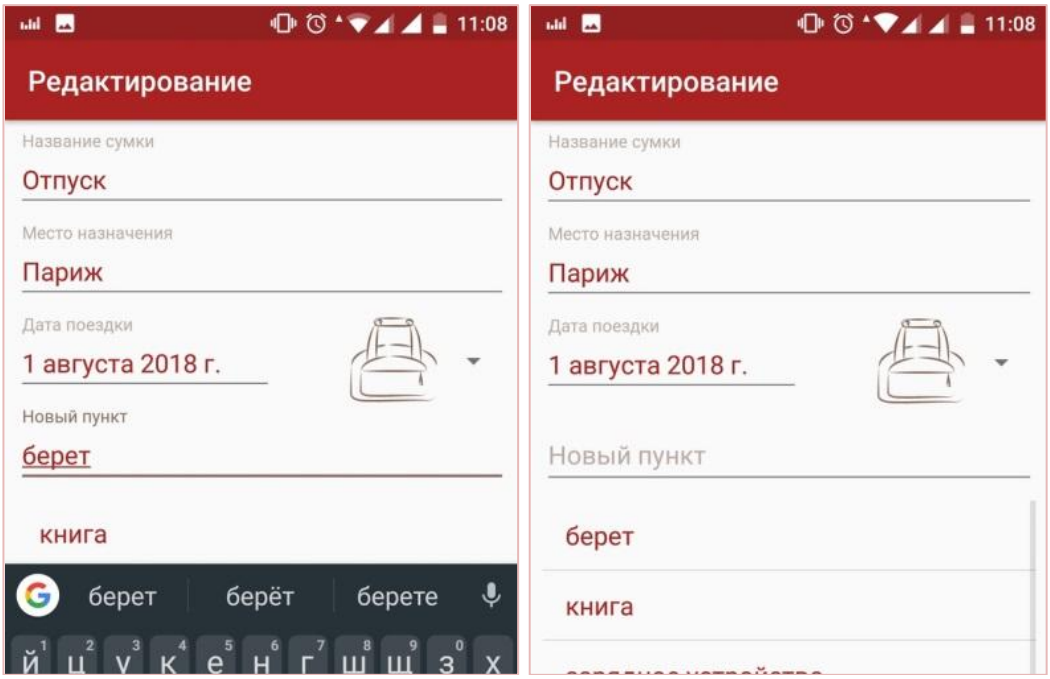


Рисунок 4.13 – Окно редактирования

Для добавления нового пункта в список нужно ввести его название в поле ввода с текстовой подсказкой «Новый пункт», после этого нажать кнопку «Enter» на клавиатуре. Новый пункт появится в начале списка (рисунок 4.14).



а) б)

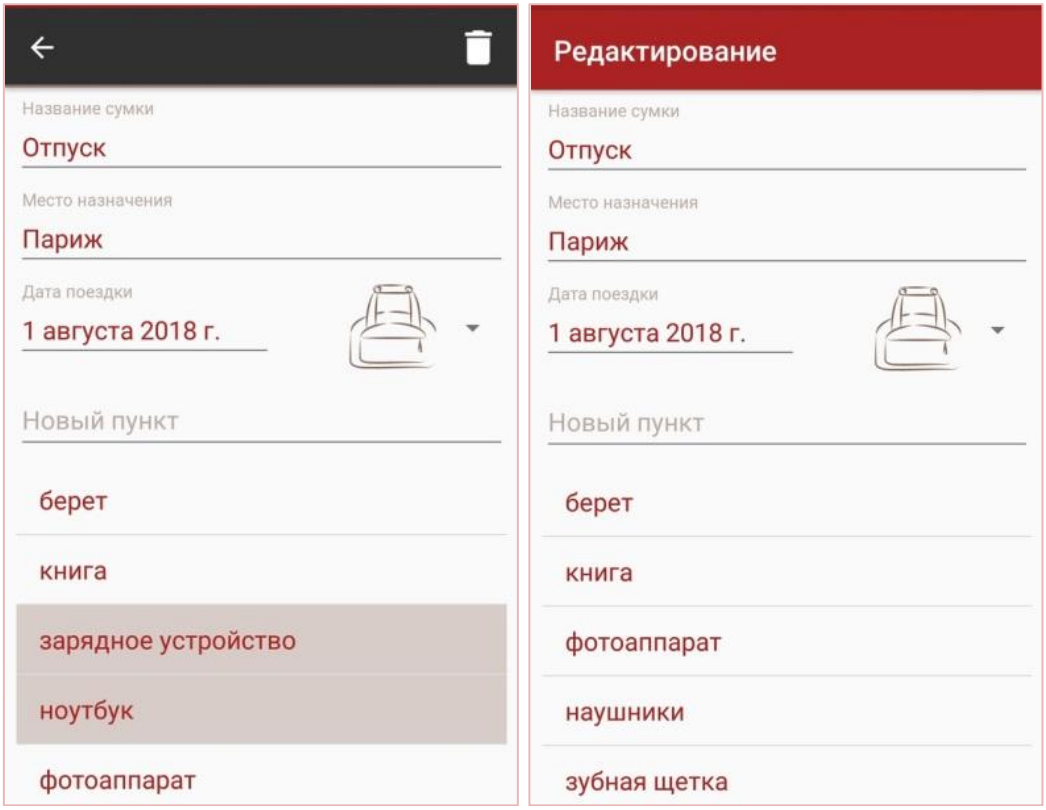
Рисунок 4.14 – Окно редактирования (1): а) демонстрация возможностей редактирования, б) добавление нового пункта в редактируемый список

Длительное нажатие на пункт списка вызовет контекстное меню, в котором будет располагаться иконка «Корзина» (рисунок 4.15).

При этом можно выделить любое количество пунктов. Для удаления выбранных пунктов нужно нажать на иконку «Корзина». Выделенные пункты удалятся, контекстное меню закроется. Чтобы выйти из контекстного меню без удаления пунктов, нужно нажать иконку «Стрелка» в верхней части экрана или снять выделение со всех пунктов путем повторного нажатия на них.

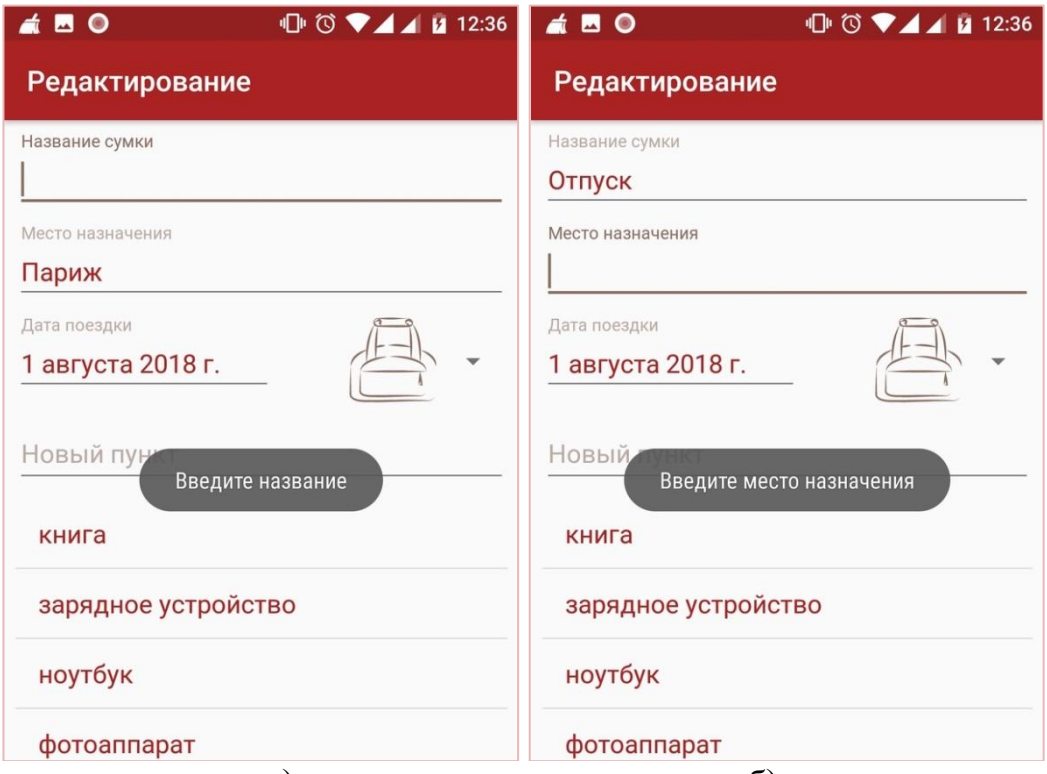
Пользователь не может покинуть окно редактирования, если не заполнено поле «Название сумки» или поле «Место назначения» (рисунок 4.16) или если он оставил список вещей пустым (рисунок 4.17).

При попытке ввода уже существующего пункта списка, как и в случае создания нового списка, будет выведено сообщение о недопустимости данного действия (рисунок 4.18).



а) б)

Рисунок 4.15 – Окно редактирования (2): а) выделение пунктов для удаления, б) список после удаления пунктов



а) б)

Рисунок 4.16 – Окно редактирования (3): а) не введено название сумки, б) не введено место назначения

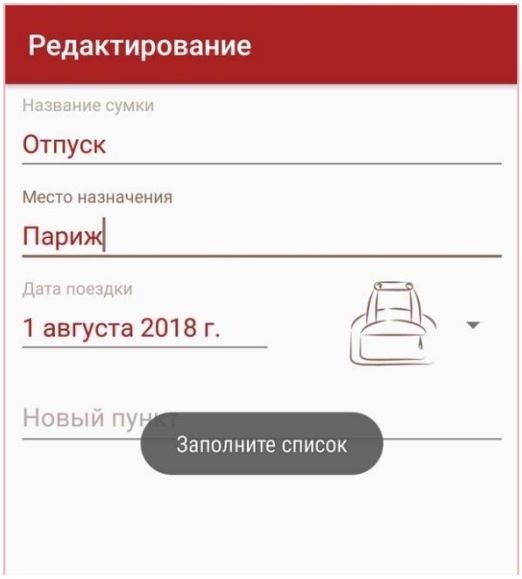


Рисунок 4.17 – Попытка возвращения в окно просмотра с пустым списком вещей

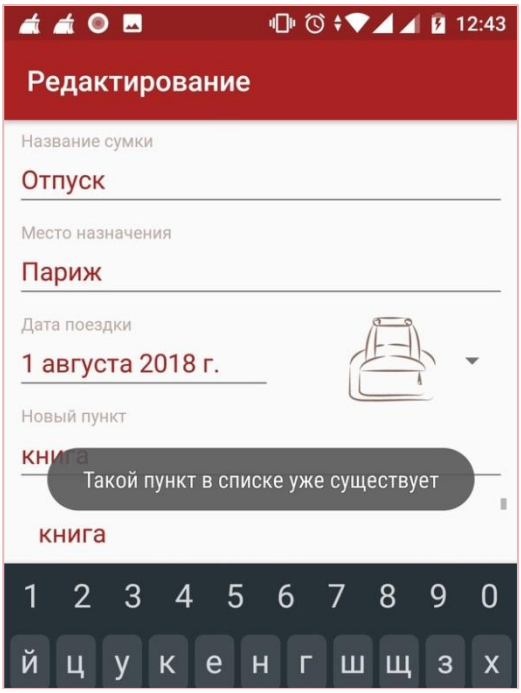


Рисунок 4.18– Попытка ввода существующего пункта в список

После завершения необходимого редактирования для выхода из данного окна необходимо нажать на кнопку «Back». Появится окно просмотра списка, а все изменения сохранятся (рисунок 4.19).

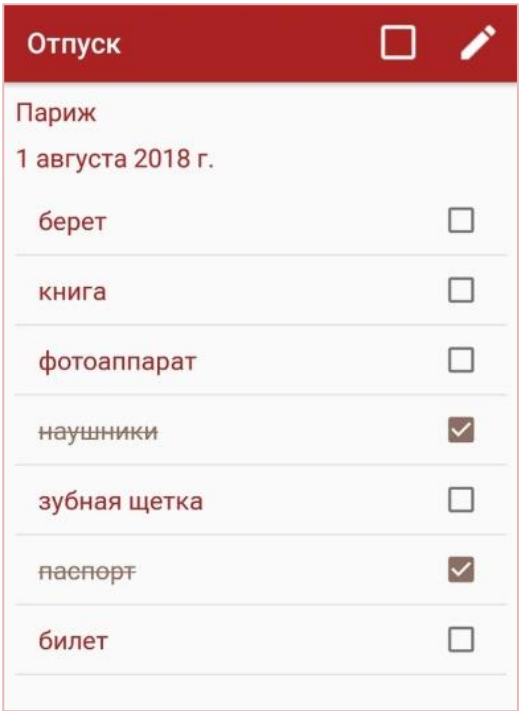
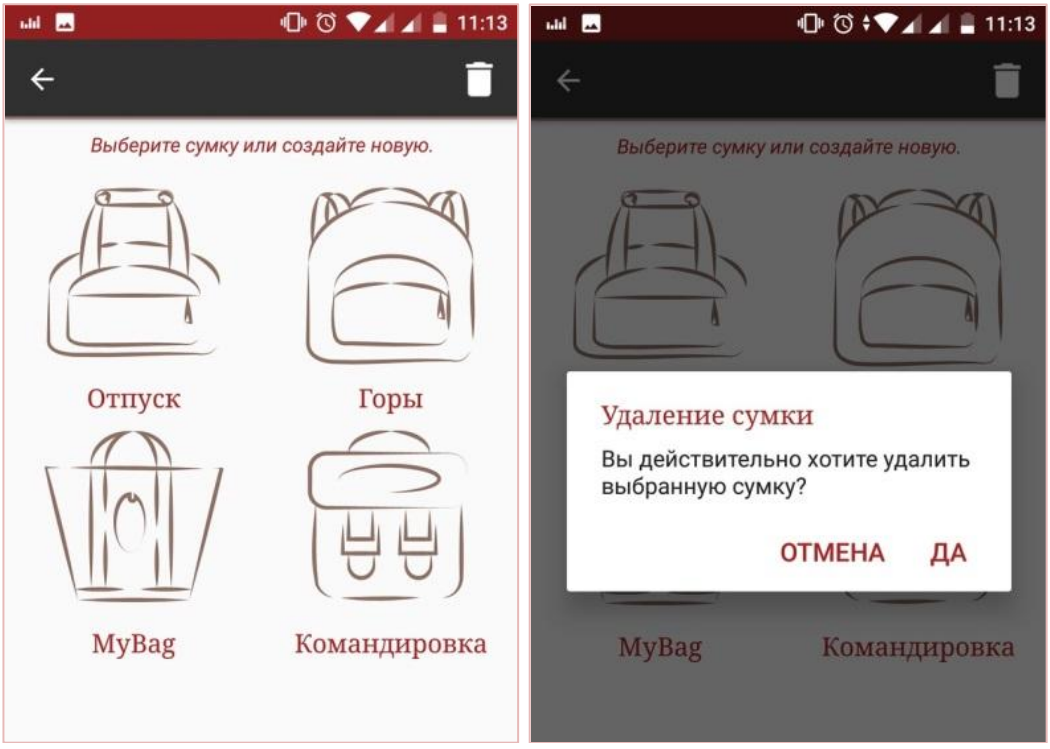


Рисунок 4.19 – Просмотр списка после редактирования

Для удаления всего списка на начальном экране необходимо длительно нажать на изображение удаляемого списка. Появится контекстное меню. В нем нужно нажать на иконку «Корзина». Появится диалоговое окно с подтверждением удаления списка (рисунок 4.20).



а) б)

Рисунок 4.20 – Удаление списка: а) вызов контекстного меню, б) подтверждение удаления

При нажатии на кнопку «Да» выбранный список удалится. При нажатии на кнопку «Отмена» контекстное меню будет закрыто (рисунок 4.21).

Рисунок 4.21 – Начальный экран после удаления списка

Для выхода из приложения нужно перейти на начальный экран и нажать кнопку «Back».

4.2 Вывод по разделу

В данном разделе был разработан пользовательский интерфейс, а также протестирована работоспособность приложения. Проведенная работа позволяет утверждать, что приложение работает корректно и имеет весь заявленный ранее функционал.

ЗАКЛЮЧЕНИЕ

В данной работе реализовано мобильное приложение «Багаж в дорогу» для системы Android. Полученное программное обеспечение удовлетворяет всем функциональным требованиям, выявленным к нему ранее в работе.

Проведена работа с потенциальными пользователями, по ходу которой были выявлены функциональные требования к программному продукту, описанные ранее.

Для визуального представления внутреннего устройства приложения созданы диаграмма классов, диаграмма вариантов использования, диаграмма компонентов. Для готового программного продукта было проведено тестирование, в ходе которого были наглядно продемонстрированы все возможности использования мобильного приложения.

В последующих версиях приложения будут добавлены следующие функциональные возможности: реализация уведомлений о предстоящих мероприятиях, создание пользовательских настроек.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Магазин приложений от компании Google. – URL: https://play.google.com/ (дата обращения: 22.02.2018).
2 Bloch, J.J. Effective Java: Programming Language Guide. — М.: Лори, 2002. — 224 с.
3 Гослинг, Дж. The Java Language Specification, Java SE 8 Edition / Дж. Гослинг, Б. Джой, Г. Стил [и др.] — М.: «Вильямс», 2015. — 672 с.
4 Академия современного программирования. – URL: http://www.amse.ru/ (дата обращения: 29.03.2018).
5 Образвательный портал IT-сферы Geekbrains. – URL: https://geekbrains.ru/ (дата обращения: 29.03.2018).
6 Хорстманн, К. С. Java. Библиотека профессионала Том 1. Основы. / К.С. Хорстманн, Г. Корнелл. – М.: «Вильямс», 2008. – 864 с.
7 Стиллмен, Э. Изучаем C#. 2-е издание. / Э. Стиллмен, Дж. Грин. — СПб.: «Питер», 2012. — 704 с.
8 Хейлсберг, А. Язык программирования C#. Классика Computers Science. 4-е издание / А. Хейлсберг, М. Торгерсен, С. Вилтамут [и др.]. — СПб.: «Питер», 2012. — 784 с.
9 Нейгел, К. C# 5.0 и платформа .NET 4.5 для профессионалов. / К. Нейгел [и др.] — М.: «Диалектика», 2013. — 1440 с.
10 Веб-ресурс, посвещенный Android-разработке. – URL: https://androidinsider.ru/ (дата обращения: 30.03.2018).
11 Сайт IT-компании AltexSoft. – URL: https://www.altexsoft.com/ (дата обращения: 04.04.2018).
12 Сайт европейской группы веб-разработчиков Netguru. – URL: https://www.netguru.co/ (дата обращения: 02.05.2018).
13 Буч, Г. Язык UML. Руководство пользователя. — 2-е изд./ Г. Буч, Дж. Рамбо, А. Джекобсон — М., СПб.: ДМК Пресс, Питер, 2004. — 432 с.
14 Учебник по разработке мобильных приложений для Android. – URL: http://startandroid.ru/ (дата обращения: 10.04.2018).
15 Сайт Александра Климова с уроками программирования. – URL: http://developer.alexanderklimov.ru/android/ (дата обращения: 12.04.2018).
16 Сайт для разработчиков Android-приложений. – URL: https://developer.android.com/ (дата обращения: 08.04.2018)

ПРИЛОЖЕНИЕ 1

ОПИСАНИЕ ПРОГРАММЫ

Багаж в дорогу

2018

ОГЛАВЛЕНИЕ

П1.1. ОБЩИЕ СВЕДЕНИЯ

Данное программное обеспечение является мобильными приложением для операционной системы Android. Написано в интегрированной среде разработки Android Studio 3.0.1 на языке программирования Java SE 7.

П1.2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

Мобильное приложение способствуют облегчению сбора личных вещей для человека перед предстоящим мероприятием, заменяя составление списков необходимых вещей на бумажном носителе.

Возможности данного программного обеспечения.

1. Создание списков личных вещей с указанием наименования списка, места назначения, даты поездки.
2. Возможность просмотра необходимого списка и реализация отметок для уже собранных вещей в списке.
3. Редактирование списка: добавление необходимых пунктов и удаление ненужных, а также редактирование данных о поездке.
4. Удаление списков.

П1.3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

Данное мобильное приложение, как и другие Android-приложения, после компиляции упаковано в один исполняемый apk-файл, который содержит в себе все файлы проекта, такие как файл манифеста (AndroidManifest.xml), dex-файл, файл ресурсов (рисунок П1.1).

Файл манифеста содержит основную информацию о программе, которая необходима операционной системе Android. Dex-файл содержит скомпилированный код приложения. Файл ресурсов включает в себя xml-файлы, в которых находятся описания макетов всех окон приложения, панели инструментов и контекстного меню, цвета, строковые константы, векторные изображения иконок, стили и т.д.

П1.4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Для функционирования программного продукта необходимо мобильное устройство, с установленной на нем операционной системой Android версии не ниже 4.0 «Ice Cream Sandwich».

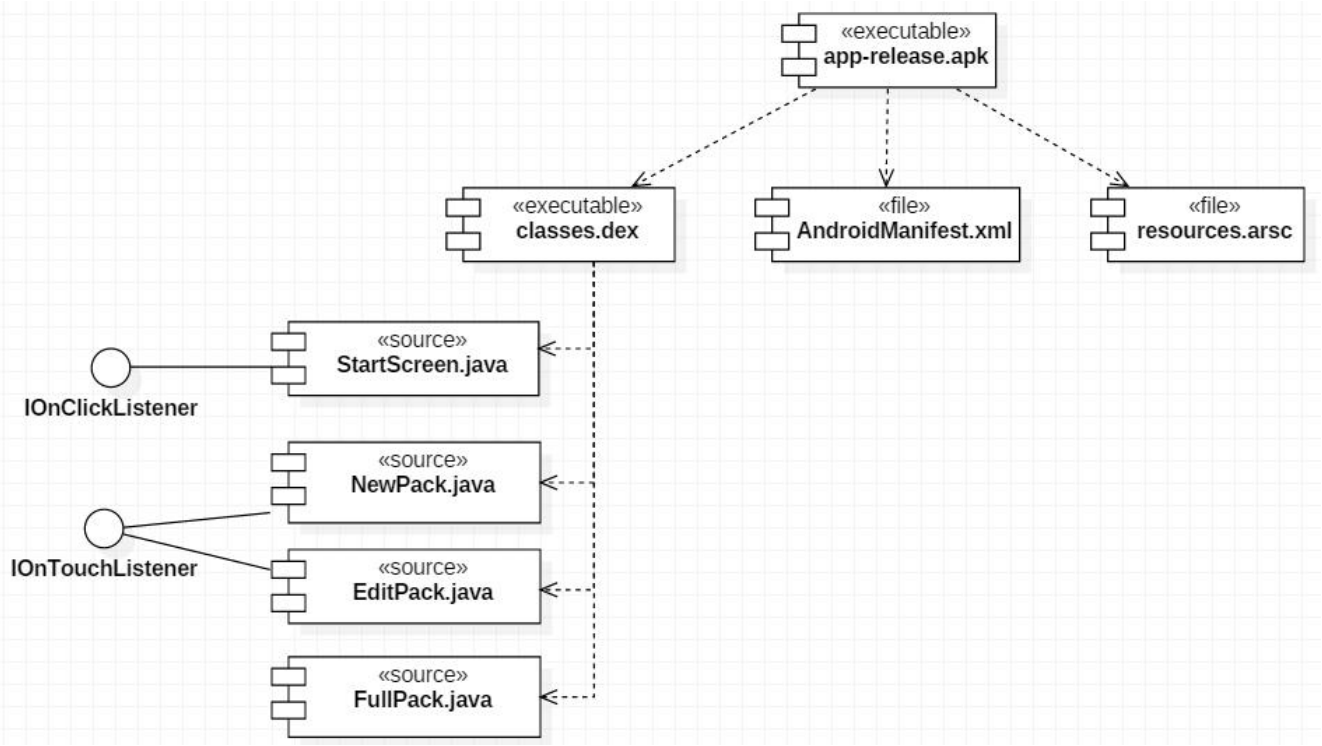


Рис. П1.1 – Диаграмма компонентов

П1.5. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Входные данные: символьные строки, вводимые пользователем в специальные редактируемые поля, выбор элементов из раскрывающихся списков, нажатие кнопок и другие касания пальцами пользователя по экрану его мобильного устройства.

Выходные данные: сохраненные данные и действия пользователя, представленные определенным образом на экране.

ПРИЛОЖЕНИЕ 2

ИСХОДНЫЙ ТЕКСТ ПРОГРАММЫ

2018

ОГЛАВЛЕНИЕ

2.1 StartScreen.java

В данном классе описан начальный экран приложения, его элементы, методы описанных элементов, чтение и запись данных в файл при запуске приложения и выходе из него.

```
   package com.example.aleksandra.easybag;

public class StartScreen extends AppCompatActivity implements
OnClickListener {

    //константы для определения activity
    private final int REQUEST_CODE_FULL_PACK = 1;
    private final int REQUEST_CODE_NEW_PACK = 2;

    private static final int NOTIFY_ID = 101; //для уведомления

    private final String LOG_TAG = "myLogs";
    private final String TAG = "States";

    private TextView topHint;
    private LinearLayout leftVerticalLayout;
    private LinearLayout rightVerticalLayout;
    private Button b2;

    private TextView newName;
    private ImageButton newIcon;

    // списки рюкзаков
    private int countPacks = 0;
    private ArrayList<Integer> icons = new ArrayList<>();
    private ArrayList<String> packNames = new ArrayList<>();
    private ArrayList<String> places = new ArrayList<>();
    private ArrayList<String>[] packLists = new ArrayList[100] ;
    private ArrayList<Integer> checkCounts  = new ArrayList<>();
    private ArrayList<Integer>[] checkKey = new ArrayList[1000];
    private ArrayList<Boolean>[] checkValue = new ArrayList[1000];

    private ArrayList<Integer> placeStates  = new ArrayList<>();

    private ArrayList<Integer> years  = new ArrayList<>();
    private ArrayList<Integer> months  = new ArrayList<>();
    private ArrayList<Integer> days  = new ArrayList<>();

    private boolean whichLayout = true;
    private int wrapContent = LinearLayout.LayoutParams.WRAP_CONTENT;
    private int matchParent = LinearLayout.LayoutParams.MATCH_PARENT;
    private LinearLayout.LayoutParams textParams = new
LinearLayout.LayoutParams(matchParent, matchParent);    //для текста
    private LinearLayout.LayoutParams iconParams = new
LinearLayout.LayoutParams(wrapContent, matchParent);   //для кнопки

    private ActionMode actionMode;
    private int clickedIconId;
```

```
    private AlertDialog.Builder alertDialog;

    private ContextThemeWrapper newContext = new ContextThemeWrapper(this,
R.style.PackName);

    @Override
    protected void onCreate(Bundle prev) {
        super.onCreate(prev);
        setContentView(R.layout.start_screen);

        Toolbar t = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(t);
        setTitle("Багаж в дорогу");

        topHint = (TextView) findViewById(R.id.topHint);
        leftVerticalLayout = (LinearLayout)
findViewById(R.id.leftVerticalLayout);
        rightVerticalLayout = (LinearLayout)
findViewById(R.id.rightVerticalLayout);
        //b2 = (Button) findViewById(R.id.button2);
        //b2.setOnClickListener(this);

        readFile();

        alertDialog = new AlertDialog.Builder(this);
        alertDialog.setTitle(R.string.alertDialogTitle);
        alertDialog.setMessage(R.string.alertDialogMessage);
        alertDialog.setPositiveButton(R.string.positiveAnswer, new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int arg1) {
                TextView deletedName;
                ImageButton deletedIcon;
                placeStates.set(clickedIconId - 1, -1);
                countPacks--;
                deletedIcon = (ImageButton)
leftVerticalLayout.findViewById(clickedIconId);
                deletedName = (TextView) leftVerticalLayout.findViewById(-2
* clickedIconId);
                if (deletedIcon == null) {
                    deletedIcon = (ImageButton)
rightVerticalLayout.findViewById(clickedIconId);
                    deletedName = (TextView)
rightVerticalLayout.findViewById(-2 * clickedIconId);
                    rightVerticalLayout.removeView(deletedIcon);
                    rightVerticalLayout.removeView(deletedName);
                } else {
                    leftVerticalLayout.removeView(deletedIcon);
                    leftVerticalLayout.removeView(deletedName);
                }
                if (countPacks == 0) topHint.setText("Создайте новый рюкзак
(иконка +).");
                actionMode.finish();
            }
        });
        alertDialog.setNegativeButton(R.string.negativeAnswer, new
```

```
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int arg1) {
                actionMode.finish();
            }
        });
        alertDialog.setCancelable(false);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.start_screen, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();

        if (id == R.id.plus) {
            Intent moveToNewPack = new Intent(this, NewPack.class);
            startActivityForResult(moveToNewPack, REQUEST_CODE_NEW_PACK);
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
        if (data == null) {return;}

        if (resultCode == RESULT_OK)
        {
            switch (requestCode) {
                case REQUEST_CODE_NEW_PACK:

                    topHint.setText("Выберите сумку или создайте новую.");
                    int position = -1;
                    for (int i = 0; i < placeStates.size(); ++i)
                    {
                        if (placeStates.get(i) == -1) {position = i;
break;}
                    }
                    int iconNumber = data.getIntExtra("iconNumber", -1);

                    //иконка нового рюкзака
                    icons.add(iconNumber);
                    Drawable icon = getIcon(iconNumber);
                    newIcon = new ImageButton(this);
                    newIcon.setImageDrawable(icon);
                    newIcon.setOnClickListener(this);
                    newIcon.setOnLongClickListener(context);

newIcon.setBackgroundColor(Color.parseColor("#00ffffff"));

                    newName = new TextView(newContext);
```

```
                    String packName = data.getStringExtra("packName");
                    packNames.add(packName);
                    newName.setText(packName);

                    String place = data.getStringExtra("place");
                    places.add(place);

                    int leftLayoutChildCount =
leftVerticalLayout.getChildCount();
                    int rightLayoutChildCount =
rightVerticalLayout.getChildCount();

                    if (leftLayoutChildCount <= rightLayoutChildCount)
                    {
                        leftVerticalLayout.addView(newIcon, iconParams);
                        leftVerticalLayout.addView(newName, textParams);
                    }
                    else
                    {
                        rightVerticalLayout.addView(newIcon, iconParams);
                        rightVerticalLayout.addView(newName, textParams);
                    }

                    if (position == -1)
                    {
                        countPacks++;
                        newIcon.setId(countPacks);
                        newName.setId(-2*countPacks);
                        //список нового рюкзака
                        packLists[countPacks-1] =
data.getStringArrayListExtra("packList");
                        placeStates.add(1);
                        years.add(data.getIntExtra("year", -1));
                        months.add(data.getIntExtra("month", -1));
                        days.add(data.getIntExtra("day", -1));
                    } else
                    {
                        newIcon.setId(position + 1);
                        newName.setId(-2*(position + 1));
                        //список нового рюкзака
                        packLists[position] =
data.getStringArrayListExtra("packList");
                        placeStates.set(position, 1);
                        years.set(position, data.getIntExtra("year", -1));
                        months.set(position, data.getIntExtra("month", -
1));
                        days.set(position, data.getIntExtra("day", -1));
                    }
                    //количество чеков (пока 0)
                    checkCounts.add(0);
                    break;

                case REQUEST_CODE_FULL_PACK:

                    //обновить данные
```

```
                    int number;
                    number = data.getIntExtra("packNumber", -1);
                    packLists[number - 1] =
data.getStringArrayListExtra("packItems");
                    checkCounts.set(number - 1,
data.getIntExtra("checkCount", -1));
                    checkKey[number - 1] = new ArrayList<>();
                    checkKey[number - 1] =
data.getIntegerArrayListExtra("checkKey");
                    boolean[] temp =
data.getBooleanArrayExtra("checkValue");
                    checkValue[number - 1] = new ArrayList<>();
                    for (int i = 0; i < temp.length; ++i)
                        checkValue[number - 1].add(temp[i]);
                    packNames.set(number - 1,
data.getStringExtra("packName"));
                    places.set(number - 1, data.getStringExtra("place"));
                    TextView tv = (TextView) findViewById(-2*number);
                    tv.setText(packNames.get(number-1));
                    iconNumber = data.getIntExtra("iconNumber", -1);
                    icons.set(number - 1, iconNumber);
                    ImageButton ib = (ImageButton) findViewById(number);
                    icon = getIcon(iconNumber);
                    ib.setImageDrawable(icon);
                    years.set(number - 1, data.getIntExtra("year", -1));
                    months.set(number - 1, data.getIntExtra("month", -1));
                    days.set(number - 1, data.getIntExtra("day", -1));
                    break;
                default:
                    break;
            }
        }
    }

    View.OnLongClickListener context = new View.OnLongClickListener() {
        @Override
        public boolean onLongClick(View v) {
            if (actionMode == null)
            {
                clickedIconId = v.getId();
                actionMode = startSupportActionMode(callback);
            }
            else
            {
                clickedIconId = 0;
                actionMode.finish();
            }
            return true;
        }
    };
    private ActionMode.Callback callback = new ActionMode.Callback() {

        public boolean onCreateActionMode(ActionMode mode, Menu menu) {
            mode.getMenuInflater().inflate(R.menu.start_screen_context,
menu);
```

```
            return true;
        }

        public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
            return false;
        }

        public boolean onActionItemClicked(ActionMode mode, MenuItem item)
{
            int id = item.getItemId();

            if (id == R.id.ic_delete) {
                alertDialog.show();
            }
            return false;
        }
        public void onDestroyActionMode(ActionMode mode) {
            actionMode = null;
        }
    };

    @Override
    public void onClick(View v) {
        int number = v.getId();
        for (int i = 0; i < countPacks; ++i) {
            if (number == i + 1) {
                Intent moveToFullPack = new Intent(this, FullPack.class);
                moveToFullPack.putStringArrayListExtra("packItems",
packLists[i]);          //список
                moveToFullPack.putExtra("packName", packNames.get(i));
//название списка
                moveToFullPack.putExtra("place", places.get(i));
                moveToFullPack.putExtra("packNumber", i + 1);
//номер рюкзака
                moveToFullPack.putExtra("checkCount", checkCounts.get(i));
//размер массива check
                moveToFullPack.putExtra("year", years.get(i));
                moveToFullPack.putExtra("month", months.get(i));
                moveToFullPack.putExtra("day", days.get(i));
                if (checkCounts.get(i) != 0) {
                    moveToFullPack.putIntegerArrayListExtra("checkKey",
checkKey[i]); //ключи check
                    boolean[] temp = new boolean[checkCounts.get(i)];
                    for (int j = 0; j < checkCounts.get(i); ++j) temp[j] =
checkValue[i].get(j);
                    moveToFullPack.putExtra("checkValue", temp);
//значения check
                }
                moveToFullPack.putExtra("iconNumber", icons.get(i));
                startActivityForResult(moveToFullPack,
REQUEST_CODE_FULL_PACK);
                break;
            }
        }
       // }
```

```
}

@Override
protected void onPause() {
    writeFile();
    super.onPause();
    Log.d(TAG, "ActivityTwo: onPause()");
}

@Override
protected void onRestart() {
    Log.d(TAG, "ActivityTwo: onRestart()");
    super.onRestart();

}
@Override
protected void onStop() {
    Log.d(TAG, "ActivityTwo: onStop()");
    writeFile();
    super.onStop();
}

@Override
protected void onDestroy() {
    Log.d(TAG, "ActivityTwo: onDestroy()");
    super.onDestroy();
    writeFile();
}

void writeFile() {
    try {
        // отрываем поток для записи
        BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(
                openFileOutput("content_main.txt", MODE_PRIVATE)));

        if (countPacks == 0) return;
        int count = 0;
        //for (int i = 0; i < placeStates.size(); ++i)
        //{
        //    if (placeStates.get(i) != -1) count++;
        //}
        bw.write(countPacks+"\n");

        for (int i = 0; i < placeStates.size(); ++i)
        {
            if (placeStates.get(i) != -1) {
                //иконка кнопки
                bw.write(icons.get(i) + "\n");
                //название рюкзака
                bw.write(packNames.get(i) + "\n");
                //место
                bw.write(places.get(i) + "\n");
                //дата
                bw.write(years.get(i) + "\n");
                bw.write(months.get(i) + "\n");
```

```
                    bw.write(days.get(i) + "\n");
                    //количество пунктвов списка
                    bw.write(packLists[i].size() + "\n");

                    //сам список
                    for (int j = 0; j < packLists[i].size(); ++j) {
                        bw.write(packLists[i].get(j) + "\n");
                    }

                    //checkbox'ы
                    if (checkCounts.size() != 0) {
                        bw.write(checkCounts.get(i) + "\n");
                        for (int j = 0; j < checkCounts.get(i); ++j) {
                            bw.write(checkKey[i].get(j) + "\n" +
checkValue[i].get(j) + "\n");
                        }
                    } else bw.write("0" + "\n");
                }
            }

            bw.close();
            Log.d(LOG_TAG, "Файл записан");
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    void readFile() {
        try {
            BufferedReader br = new BufferedReader(new InputStreamReader(
                    openFileInput("content_main.txt")));
            String str = "";
               if ((str = br.readLine()) != null) countPacks =
Integer.parseInt(str);
            else
            {
                topHint.setText("Создайте новый рюкзак (иконка +).");
                return;
            }
            topHint.setText("Выберите сумку или создайте новую.");
            icons = new ArrayList<>(countPacks);
            packNames = new ArrayList<>(countPacks);
            places = new ArrayList<>(countPacks);
            years = new  ArrayList<>(countPacks);
            months = new  ArrayList<>(countPacks);
            days = new  ArrayList<>(countPacks);
            checkCounts = new ArrayList<>(countPacks);
            placeStates = new ArrayList<>(countPacks);
            for(int i = 0; i < countPacks; ++i)
            {
                icons.add(Integer.parseInt(br.readLine()));
                placeStates.add(1);
                str = br.readLine();
                newName = new TextView(newContext);
```

```
                packNames.add(str);
                newName.setText(str);

                places.add(br.readLine());

                Drawable icon = getIcon(icons.get(i));
                newIcon = new ImageButton(this);
                newIcon.setImageDrawable(icon);
                newIcon.setId(i + 1);
                newIcon.setOnClickListener(this);
                newIcon.setOnLongClickListener(context);
                newIcon.setBackgroundColor(Color.parseColor("#00ffffff"));

                newName.setId(-2 * (i + 1));
                if (whichLayout)
                {
                    leftVerticalLayout.addView(newIcon, iconParams);
                    leftVerticalLayout.addView(newName, textParams);
                }
                else
                {
                    rightVerticalLayout.addView(newIcon, iconParams);
                    rightVerticalLayout.addView(newName, textParams);
                }
                whichLayout = whichLayout == true? false: true;

                years.add(Integer.parseInt(br.readLine()));
                months.add(Integer.parseInt(br.readLine()));
                days.add(Integer.parseInt(br.readLine()));

                int sizeArrayList = Integer.parseInt(br.readLine());
                packLists[i] = new ArrayList<>(sizeArrayList);
                for (int j = 0; j < sizeArrayList; ++j)
                {
                    packLists[i].add(br.readLine());
                }
                checkCounts.add(Integer.parseInt(br.readLine()));
                if(checkCounts.get(i) != 0)
                {
                    checkKey[i] = new ArrayList<>(checkCounts.get(i));
                    checkValue[i] = new ArrayList<>(checkCounts.get(i));
                    for (int j = 0; j < checkCounts.get(i); ++j)
                    {
                        checkKey[i].add(Integer.parseInt(br.readLine()));
                        str = br.readLine();
                        if (str.equals("false")) checkValue[i].add(false);
                        else if (str.equals("true"))
checkValue[i].add(true);
                    }
                }
            }

        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }

    private Drawable getIcon(int iconNumber) {
        Resources res = getResources();
        Drawable icon = res.getDrawable(R.drawable.ic_bag);
        switch (iconNumber)
        {
            case 0:
                icon = res.getDrawable(R.drawable.ic_bag);
                break;
            case 1:
                icon = res.getDrawable(R.drawable.ic_bag2);
                break;
            case 2:
                icon = res.getDrawable(R.drawable.ic_bag3);
                break;
            case 3:
                icon = res.getDrawable(R.drawable.ic_bag4);
                break;
            case 4:
                icon = res.getDrawable(R.drawable.ic_bag5);
                break;
        }
        return icon;
    }

}
```

2.2 NewPack.java

В данном классе описано окно создания нового списка вещей, а также механизм передачи и сохранения введенной информации на начальный кран приложения.

```
   package com.example.aleksandra.easybag;

public class NewPack extends AppCompatActivity implements OnTouchListener {

    private TextInputEditText name;
    private TextInputEditText dateText;
    private TextInputEditText item;
    private TextInputEditText place;
    private ListView pack;
    private Spinner spinner;

    private int countItem = 0;

    private ArrayList<String> packItems = new ArrayList<>();
    private ArrayList<Boolean> selectedItems = new ArrayList<>();

    private Calendar date = Calendar.getInstance();
    private int year = Calendar.YEAR;
    private int month = Calendar.MONTH;
```

```
    private int day = Calendar.DAY_OF_MONTH;

    @Override
    protected void onCreate(Bundle prev) {
        super.onCreate(prev);
        setContentView(R.layout.new_pack);
        Toolbar t = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(t);
        setTitle("Новая сумка");

        name = (TextInputEditText) findViewById(R.id.nameNewPack);
        dateText = (TextInputEditText) findViewById(R.id.date);
        dateText.setOnTouchListener(this);
        item = (TextInputEditText) findViewById(R.id.newItem);
        place = (TextInputEditText) findViewById(R.id.newPlace);
        pack = (ListView) findViewById(R.id.newPack);
        pack.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE_MODAL);
        spinner = (Spinner) findViewById(R.id.spinner);

        IconArrayAdapter spinnerAdapter = new IconArrayAdapter(this,
                new Integer[]{  R.drawable.ic_bag_64,
R.drawable.ic_bag2_64,
                                R.drawable.ic_bag3_64,
R.drawable.ic_bag4_64,
                                R.drawable.ic_bag5_64    });
        spinner.setAdapter(spinnerAdapter);

        final ArrayAdapter<String> adapter = new ReverseAdapter(this);
        pack.setAdapter(adapter);
        pack.setMultiChoiceModeListener(new
AbsListView.MultiChoiceModeListener() {

            @Override
            public boolean onCreateActionMode(android.view.ActionMode mode,
Menu menu) {
                mode.getMenuInflater().inflate(R.menu.new_pack_context,
menu);
                return true;
            }

            @Override
            public boolean onPrepareActionMode(android.view.ActionMode
actionMode, Menu menu) { return false; }

            @Override
            public boolean onActionItemClicked(android.view.ActionMode
mode, MenuItem menuItem) {
                int id = menuItem.getItemId();
                if (id == R.id.deleteNewItems)
                {
                    for (int i = 0; i < packItems.size(); ++i)
                        if (selectedItems.get(i))
                        {
                            packItems.remove(packItems.size() - i - 1);
                            selectedItems.remove(i);
```

```
                            i--;
                        }
                    }
                    mode.finish();
                    return false;
                }

                @Override
                public void onDestroyActionMode(android.view.ActionMode
actionMode) {
                    for (int i = 0; i < pack.getChildCount(); ++i) {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.backg
round));
                    }
                }

                @Override
                public void onItemCheckedStateChanged(android.view.ActionMode
actionMode, int i, long l, boolean b) {
                    selectedItems.set(i, b);
                    if (b) {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.light
Brown));

                    }
                    else {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.backg
round));
                    }
                }
            });

            item.setOnKeyListener(new View.OnKeyListener() {
                public boolean onKey(View v, int keyCode, KeyEvent event) {
                    if (event.getAction() == KeyEvent.ACTION_DOWN)
                        if (keyCode == KeyEvent.KEYCODE_ENTER) {
                            if (item.getText().toString().equals("")) return
false;
                            if (packItems.contains(item.getText().toString()))
{
                                Toast toast =
Toast.makeText(getApplicationContext(), "Такой пункт в списке уже
существует", Toast.LENGTH_SHORT);
                                toast.setGravity(Gravity.CENTER, 0, 0);
                                toast.show();
                                return false;
                            }
                            countItem++;
                            packItems.add(packItems.size(),
item.getText().toString());
                            selectedItems.add(false);
                            //обновление списка
```

```
                        adapter.notifyDataSetChanged();
                        //pack.smoothScrollToPosition(packItems.size()+1);
                        item.setText("");
                        //фокус на editText и show keyboard
                        item.post(new Runnable(){
                            @Override
                            public void run()
                            {
                                item.requestFocus();
                                InputMethodManager imm =
(InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
                                imm.showSoftInput(item,
InputMethodManager.SHOW_IMPLICIT);
                            }
                        });
                        return true;
                    }
                return false;
            }
        });


    }

    @Override
    public boolean onTouch(View v, MotionEvent event) {

        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN: // нажатие
                setDate(v);
                break;
            case MotionEvent.ACTION_MOVE: // движение
                setDate(v);
                break;
            case MotionEvent.ACTION_UP: // отпускание
            case MotionEvent.ACTION_CANCEL:
                break;
        }
        return true;
    }

    public void setDate(View v) {
        new DatePickerDialog(NewPack.this, d,
                date.get(Calendar.YEAR),
                date.get(Calendar.MONTH),
                date.get(Calendar.DAY_OF_MONTH))
                .show();
    }
    private void setInitialDate() {
        dateText.setText(DateUtils.formatDateTime(this,
                date.getTimeInMillis(),
                DateUtils.FORMAT_SHOW_DATE | DateUtils.FORMAT_SHOW_YEAR));
    }
    DatePickerDialog.OnDateSetListener d = new
DatePickerDialog.OnDateSetListener() {
```

```
        public void onDateSet(DatePicker view, int y, int monthOfYear, int
dayOfMonth) {
            year = y;
            month = monthOfYear;
            day = dayOfMonth;
            date.set(Calendar.YEAR, y);
            date.set(Calendar.MONTH, monthOfYear);
            date.set(Calendar.DAY_OF_MONTH, dayOfMonth);
            setInitialDate();
        }
    };

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.new_pack, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {

        int id = item.getItemId();

        if (id == R.id.ic_save) {
            if (name.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Введите
название",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (packItems.size() == 0 &&
this.item.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Заполните
список",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (place.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Введите место
назначения",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (dateText.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Укажите дату
поездки",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
```

```
                return false;
            }
            if(!this.item.getText().toString().equals(""))
            {
                countItem++;
                packItems.add(packItems.size(),
this.item.getText().toString());
            }
            int position = spinner.getSelectedItemPosition();
            Intent backToStartScreen = new Intent();
            backToStartScreen.putExtra("iconNumber", position);
            backToStartScreen.putExtra("year", year);
            backToStartScreen.putExtra("month", month);
            backToStartScreen.putExtra("day", day);
            backToStartScreen.putExtra("packName",
name.getText().toString());
            backToStartScreen.putExtra("place",
place.getText().toString());
            backToStartScreen.putStringArrayListExtra("packList",
packItems);
            setResult(RESULT_OK, backToStartScreen);
            finish();
        }
        return super.onOptionsItemSelected(item);
    }
    private class ReverseAdapter extends ArrayAdapter<String> {

        private ReverseAdapter(Context context) {
            super(context, android.R.layout.simple_list_item_activated_1,
packItems);
        }

        @Override
        public String getItem(int position) {
            return packItems.get(packItems.size() - position - 1);
        }
    }

    private class IconArrayAdapter extends ArrayAdapter<Integer> {
       private Integer[] images;

       private IconArrayAdapter(Context context, Integer[] images) {
           super(context, android.R.layout.select_dialog_item, images);
           this.images = images;
       }

       @Override
       public View getDropDownView(int position, View convertView,
ViewGroup parent) {
           return getImageForPosition(position);
       }

       @Override
       public View getView(int position, View convertView, ViewGroup
parent) {
```

```
            return getImageForPosition(position);
        }

        private View getImageForPosition(int position) {
            ImageView imageView = new ImageView(getContext());
            imageView.setBackgroundResource(images[position]);
            imageView.setLayoutParams(new
AbsListView.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT));
            return imageView;
        }
    }

}
```

2.3 FullPack.java

В данном классе описано окно просмотра списка вещей, а также механизм передачи информации между начальным экраном приложения и окном редактирования.

```
   package com.example.aleksandra.easybag;

public class FullPack extends AppCompatActivity {

    final int REQUEST_CODE_EDIT_PACK = 3;

    private ListView pack;
    private TextView placeInfo;
    private TextView dateInfo;

    private ArrayList<String> packItems = new ArrayList<>();
    private ArrayList<Integer> checkKey = new ArrayList<>();
    private boolean[] checkValue;
    private String packName;
    private String place;
    private int packNumber;
    private int checkCount;
    private int iconNumber;
    private int year;
    private int month;
    private int day;
    private Calendar date = Calendar.getInstance();

    private CheckListAdapter adapter;

    private boolean whichIcon = false;
    private Menu menu;

    @Override
    protected void onCreate(Bundle prev) {
        super.onCreate(prev);

        Bundle extras = getIntent().getExtras();
        packItems = extras.getStringArrayList("packItems");
```

```
        checkCount = extras.getInt("checkCount");
        if (checkCount != 0)
        {
            checkValue = new boolean[checkCount];
            checkKey = extras.getIntegerArrayList("checkKey");
            checkValue = extras.getBooleanArray("checkValue");
        }
        packName = extras.getString("packName");
        place = extras.getString("place");
        packNumber = extras.getInt("packNumber");
        iconNumber = extras.getInt("iconNumber");
        year = extras.getInt("year");
        month = extras.getInt("month");
        day = extras.getInt("day");

        setContentView(R.layout.full_pack);
        Toolbar t = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(t);
        setTitle(packName);

        pack = (ListView) findViewById(R.id.fullPack);
        pack.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
        placeInfo = (TextView) findViewById(R.id.placeInfo);
        placeInfo.setText(place);
        dateInfo = (TextView) findViewById(R.id.dateInfo);
        date.set(Calendar.YEAR, year);
        date.set(Calendar.MONTH, month);
        date.set(Calendar.DAY_OF_MONTH, day);
        setInitialDate();

        adapter = new CheckListAdapter();
        pack.setAdapter(adapter);

        //восстанавливаем checkbox'ы
        if(checkCount != 0)
        {
            for (int i = 0; i < checkCount; ++i) {
                if (checkValue[i])
                    pack.setItemChecked(checkKey.get(i), true);
            }
        }
        adapter.notifyDataSetChanged();

        pack.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view,
int i, long l) {
                //если пункт уже вычеркнут

if(adapter.getSelectedStrings().contains(adapter.getItem(i)))
                {
                    //восстанавливаем его

adapter.getSelectedStrings().remove(adapter.getItem(i));
                }
```

```
                //иначе зачеркиваем
                else
adapter.getSelectedStrings().add(packItems.get(packItems.size() - i - 1));
                adapter.notifyDataSetChanged();
            }
        });

    }

    private void setInitialDate() {
        dateInfo.setText(DateUtils.formatDateTime(this,
                date.getTimeInMillis(),
                DateUtils.FORMAT_SHOW_DATE | DateUtils.FORMAT_SHOW_YEAR));
    }

    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_BACK){
            //key-booleanValue
            SparseBooleanArray selectedItems =
pack.getCheckedItemPositions();
            checkCount = 0;
            checkKey = new ArrayList<>();
            for (int i = 0; i < packItems.size(); i++) {
                if(selectedItems.get(i))
                {
                    checkKey.add(i);
                    checkCount++;
                }
            }
            checkValue = new boolean[checkCount];
            for (int i = 0; i < checkCount; i++) { checkValue[i] = true; }
            Intent backToMainScreen = new Intent();
            backToMainScreen.putExtra("checkCount", checkCount);
            backToMainScreen.putIntegerArrayListExtra("checkKey",
checkKey);
            backToMainScreen.putExtra("checkValue", checkValue);
            backToMainScreen.putExtra("packNumber", packNumber);
            backToMainScreen.putStringArrayListExtra("packItems",
packItems);
            backToMainScreen.putExtra("packName", packName);
            backToMainScreen.putExtra("place", place);
            backToMainScreen.putExtra("iconNumber", iconNumber);
            backToMainScreen.putExtra("year", year);
            backToMainScreen.putExtra("month", month);
            backToMainScreen.putExtra("day", day);
            setResult(RESULT_OK, backToMainScreen);
            finish();
            return true;
        }
        return super.onKeyDown(keyCode, event);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
```

```
        getMenuInflater().inflate(R.menu.full_pack, menu);
        this.menu = menu;
        SparseBooleanArray selectedItems = pack.getCheckedItemPositions();
        if(checkCount !=0) {
            if (selectedItems.size() == packItems.size()) {

this.menu.getItem(0).setIcon(ContextCompat.getDrawable(this,
R.drawable.checkbox_checked_white));
                whichIcon = !whichIcon;
            }
        }
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();

        //Редактирование списка
        if (id == R.id.edit) {
            SparseBooleanArray selectedItems =
pack.getCheckedItemPositions();
            checkKey = new ArrayList<>();
            checkValue = new boolean[packItems.size()];
            for (int i = 0; i < packItems.size(); i++) {
                if(selectedItems.get(i))
                {
                    checkKey.add(i);
                    checkValue[i] = true;

                }else
                {
                    checkKey.add(i);
                    checkValue[i] = false;
                }
            }
            Intent moveToEditPack = new Intent(this, EditPack.class);
            moveToEditPack.putIntegerArrayListExtra("checkKey", checkKey);
            moveToEditPack.putExtra("checkValue", checkValue);
            moveToEditPack.putStringArrayListExtra("packItems", packItems);
            moveToEditPack.putExtra("packName", packName);
            moveToEditPack.putExtra("place", place);
            moveToEditPack.putExtra("iconNumber", iconNumber);
            moveToEditPack.putExtra("year", year);
            moveToEditPack.putExtra("month", month);
            moveToEditPack.putExtra("day", day);
            moveToEditPack.putExtra("checkCount", checkCount);
            startActivityForResult(moveToEditPack, REQUEST_CODE_EDIT_PACK);
        }
        if (id == R.id.check) {
            ActionMenuItemView menuItem = (ActionMenuItemView)
findViewById(R.id.check);
            checkValue = new boolean[packItems.size()];
            checkKey = new ArrayList<>();
            if (whichIcon){
```

```
                for (int i = 0; i < packItems.size(); ++i) {
                    pack.setItemChecked(i, false);
                    checkValue[i] = false;
                    checkKey.add(i);
                    //если пункт уже вычеркнут

if(adapter.getSelectedStrings().contains(adapter.getItem(i)))
                    {
                        //восстанавливаем его

adapter.getSelectedStrings().remove(adapter.getItem(i));
                    }
                }
                menu.getItem(0).setIcon(ContextCompat.getDrawable(this,
R.drawable.checkbox_white));
                whichIcon = !whichIcon;
            }
            else {
                for (int i = 0; i < packItems.size(); ++i) {
                    pack.setItemChecked(i, true);
                    checkValue[i] = true;
                    checkKey.add(i);
                    //если пункт уже вычеркнут

if(adapter.getSelectedStrings().contains(adapter.getItem(i)))
                    {

                    }
                    //иначе зачеркиваем
                    else
adapter.getSelectedStrings().add(packItems.get(packItems.size() - i - 1));

                }
                menu.getItem(0).setIcon(ContextCompat.getDrawable(this,
R.drawable.checkbox_checked_white));
                whichIcon = !whichIcon;
            }
            adapter.notifyDataSetChanged();

        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
        if (data == null) {return;}

        if (resultCode == RESULT_OK) {
            switch (requestCode) {
                case REQUEST_CODE_EDIT_PACK:
                    adapter = new CheckListAdapter();
                    pack.setAdapter(adapter);
                    packItems = data.getStringArrayListExtra("packItems");
                    checkKey = data.getIntegerArrayListExtra("checkKey");
```

```
                    checkValue = new boolean[packItems.size()];
                    checkValue = data.getBooleanArrayExtra("checkValue");
                    for (int i = 0; i < packItems.size(); ++i) {
                        if (checkValue[i])
                            pack.setItemChecked(checkKey.get(i), true);
                        else pack.setItemChecked(checkKey.get(i), false);
                    }

                    adapter.notifyDataSetChanged();
                    packName = data.getStringExtra("packName");
                    place = data.getStringExtra("place");
                    placeInfo.setText(place);
                    setTitle(packName);
                    iconNumber = data.getIntExtra("iconNumber", -1);
                    year = data.getIntExtra("year", -1);
                    month = data.getIntExtra("month", -1);
                    day = data.getIntExtra("day", -1);
                    date.set(Calendar.YEAR, year);
                    date.set(Calendar.MONTH, month);
                    date.set(Calendar.DAY_OF_MONTH, day);
                    setInitialDate();
                }
            }
        }

        private class CheckListAdapter extends BaseAdapter {

            private ArrayList<String> selectedStrings;

            private CheckListAdapter() {
                selectedStrings = new ArrayList<>();
                if (checkValue != null)
                {
                    for (int i = 0; i < checkValue.length; ++i) {
                        if (checkValue[i])
                            selectedStrings.add(packItems.get(packItems.size()
- checkKey.get(i) - 1));
                    }
                }
            }

            @Override
            public int getCount() {
                return packItems.size();
            }

            @Override
            public String getItem(int position) {
                return packItems.get(packItems.size() - position - 1);
            }

            @Override
            public long getItemId(int i) {
                return 0;
            }
```

```
        @Override
        public View getView(int i, View view, ViewGroup viewGroup) {
            ViewHolder holder;
            if (view == null) {
                holder = new ViewHolder();
                view =
LayoutInflater.from(viewGroup.getContext()).inflate(android.R.layout.simple
_list_item_multiple_choice, viewGroup, false);
                holder.text = (TextView) view;
                view.setTag(holder);
            } else
                holder = (ViewHolder) view.getTag();
            holder.text.setText(getItem(i));
            if (selectedStrings.contains(getItem(i))) {
                holder.text.setPaintFlags(holder.text.getPaintFlags() |
Paint.STRIKE_THRU_TEXT_FLAG);

holder.text.setTextColor(getResources().getColor(R.color.colorAccent));
            }
            else{
                holder.text.setPaintFlags(0);

holder.text.setTextColor(getResources().getColor(R.color.colorPrimary));
            }
            return view;
        }

        private class ViewHolder {
            TextView text;
        }

        private ArrayList<String> getSelectedStrings() {
            return selectedStrings;
        }
    }

}
```

2.4 EditPack.java

В данном классе описано окно редактирования, а также сохранение и передача измененной информации в окно просмотра списка.

```
   package com.example.aleksandra.easybag;

public class EditPack extends AppCompatActivity implements OnTouchListener{

    private TextInputEditText packName;
    private TextInputEditText newItem;
    private TextInputEditText editPlace;
    private ListView pack;
    private Spinner editSpinner;

    private ArrayList<String> packItems = new ArrayList<>();
```

```
    private ArrayList<Integer> checkKey = new ArrayList<>();
    private boolean[] checkValue = new boolean[1000];
    private ArrayList<Integer> checkVal = new ArrayList<>();
    private ArrayList<Boolean> selectedItems = new ArrayList<>();
    private int iconNumber;

    private TextInputEditText dateText;
    private Calendar date = Calendar.getInstance();;
    private int year;
    private int month;
    private int day;

    private String name;
    private String place;

    @Override
    protected void onCreate(Bundle prev) {
        super.onCreate(prev);

        Bundle extras = getIntent().getExtras();
        packItems = extras.getStringArrayList("packItems");
        for(int i = 0; i < packItems.size(); ++i)
            selectedItems.add(false);
        checkValue = new boolean[packItems.size()];
        checkValue = new boolean[packItems.size()];
        checkKey = extras.getIntegerArrayList("checkKey");
        checkValue = extras.getBooleanArray("checkValue");
        for (int i = 0; i < packItems.size(); ++i)
        {
            if (checkValue[i]) checkVal.add(1);
            else checkVal.add(0);
        }
        name = extras.getString("packName");
        place = extras.getString("place");
        iconNumber = extras.getInt("iconNumber");
        year = extras.getInt("year");
        month = extras.getInt("month");
        day = extras.getInt("day");

        setContentView(R.layout.edit_pack);
        Toolbar t = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(t);
        setTitle("Редактирование");

        packName = (TextInputEditText) findViewById(R.id.editNamePack);
        packName.setText(name);
        newItem = (TextInputEditText) findViewById(R.id.editNewItem);
        editPlace = (TextInputEditText) findViewById(R.id.editPlace);
        editPlace.setText(place);
        pack = (ListView) findViewById(R.id.editPack);
        pack.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE_MODAL);
        editSpinner = (Spinner) findViewById(R.id.editSpinner);

        EditPack.IconArrayAdapter spinnerAdapter = new
EditPack.IconArrayAdapter(this,
```

```
                new Integer[]{  R.drawable.ic_bag_64,
R.drawable.ic_bag2_64,
                        R.drawable.ic_bag3_64, R.drawable.ic_bag4_64,
                        R.drawable.ic_bag5_64  });
        editSpinner.setAdapter(spinnerAdapter);
        editSpinner.setSelection(iconNumber);

        final ArrayAdapter<String> adapter = new
EditPack.ReverseAdapter(this);
        pack.setAdapter(adapter);

        pack.setMultiChoiceModeListener(new
AbsListView.MultiChoiceModeListener() {

            @Override
            public boolean onCreateActionMode(android.view.ActionMode mode,
Menu menu) {
                mode.getMenuInflater().inflate(R.menu.edit_pack_context,
menu);
                return true;
            }

            @Override
            public boolean onPrepareActionMode(android.view.ActionMode
actionMode, Menu menu) { return false; }

            @Override
            public boolean onActionItemClicked(android.view.ActionMode
mode, MenuItem menuItem) {
                int id = menuItem.getItemId();
                if (id == R.id.deleteItems)
                {
                    for (int i = 0; i < packItems.size(); ++i)
                        if (selectedItems.get(i))
                        {
                            packItems.remove(packItems.size() - i - 1);
                            selectedItems.remove(i);
                            checkVal.remove(i);
                            checkKey.remove(i);
                            for (int j = i; j < packItems.size(); ++j)
                                checkKey.set(j, j);
                            i--;
                        }
                }
                mode.finish();
                return false;
            }

            @Override
            public void onDestroyActionMode(android.view.ActionMode
actionMode) {
                for (int i = 0; i < pack.getChildCount(); ++i) {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.backg
round));
```

```
                }
            }

            @Override
            public void onItemCheckedStateChanged(android.view.ActionMode
actionMode, int i, long l, boolean b) {
                selectedItems.set(i, b);
                if (b) {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.light
Brown));

                }
                else {

pack.getChildAt(i).setBackgroundColor(getResources().getColor(R.color.backg
round));
                }
            }
        });

        newItem.setOnKeyListener(new View.OnKeyListener() {
            public boolean onKey(View v, int keyCode, KeyEvent event) {
                if (event.getAction() == KeyEvent.ACTION_DOWN)
                    if (keyCode == KeyEvent.KEYCODE_ENTER) {
                        if
(packItems.contains(newItem.getText().toString())) {
                            Toast toast =
Toast.makeText(getApplicationContext(), "Такой пункт в списке уже
существует", Toast.LENGTH_SHORT);
                            toast.setGravity(Gravity.CENTER, 0, 0);
                            toast.show();
                            return false;
                        }
                        if (!newItem.getText().toString().equals(""))
                        {
                            //скрытие клавиатуры
                            InputMethodManager imm = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);

imm.hideSoftInputFromWindow(newItem.getWindowToken(), 0);
                            packItems.add(newItem.getText().toString());
                            adapter.notifyDataSetChanged();
                            newItem.setText("");
                            selectedItems.add(false);
                            checkVal.add(0);
                            checkKey.add(packItems.size() - 1);
                            return true;
                        }
                    }
                return false;
            }
        });

        dateText = (TextInputEditText) findViewById(R.id.editDate);
```

```
        dateText.setOnTouchListener(this);
        date.set(Calendar.YEAR, year);
        date.set(Calendar.MONTH, month);
        date.set(Calendar.DAY_OF_MONTH, day);
        setInitialDate();
    }

    public void setDate(View v) {
        new DatePickerDialog(EditPack.this, d,
                date.get(Calendar.YEAR),
                date.get(Calendar.MONTH),
                date.get(Calendar.DAY_OF_MONTH))
                .show();
    }

    private void setInitialDate() {
        dateText.setText(DateUtils.formatDateTime(this,
                date.getTimeInMillis(),
                DateUtils.FORMAT_SHOW_DATE | DateUtils.FORMAT_SHOW_YEAR));
    }
    DatePickerDialog.OnDateSetListener d = new
DatePickerDialog.OnDateSetListener() {
        public void onDateSet(DatePicker view, int y, int monthOfYear, int
dayOfMonth) {
            year = y;
            month = monthOfYear;
            day = dayOfMonth;
            date.set(Calendar.YEAR, y);
            date.set(Calendar.MONTH, monthOfYear);
            date.set(Calendar.DAY_OF_MONTH, dayOfMonth);
            setInitialDate();
        }
    };

    @Override
    public boolean onTouch(View v, MotionEvent event) {

        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN: // нажатие
                setDate(v);
                break;
            case MotionEvent.ACTION_MOVE: // движение
                setDate(v);
                break;
            case MotionEvent.ACTION_UP: // отпускание
            case MotionEvent.ACTION_CANCEL:
                break;
        }
        return true;
    }

    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_BACK){
            checkValue = new boolean[packItems.size()];
```

```
            if (packName.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Введите
название",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (packItems.size() == 0 &&
this.newItem.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Заполните
список",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (editPlace.getText().toString().equals("")){
                Toast toast =
Toast.makeText(getApplicationContext(),"Введите место
назначения",Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            if (dateText.getText().toString().equals("")) {
                Toast toast = Toast.makeText(getApplicationContext(),
"Укажите дату поездки", Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
                return false;
            }
            for (int i = 0; i < packItems.size(); ++i)
                if (checkVal.get(packItems.size() - i - 1) == 1)
checkValue[i] = true;
                else checkValue[i] = false;
            Intent backToFullPack = new Intent();
            backToFullPack.putIntegerArrayListExtra("checkKey", checkKey);
            backToFullPack.putExtra("checkValue", checkValue);
            backToFullPack.putStringArrayListExtra("packItems", packItems);
            name = packName.getText().toString();
            backToFullPack.putExtra("packName", name);
            place = editPlace.getText().toString();
            backToFullPack.putExtra("place", place);
            iconNumber = editSpinner.getSelectedItemPosition();
            backToFullPack.putExtra("iconNumber", iconNumber);
            backToFullPack.putExtra("year", year);
            backToFullPack.putExtra("month", month);
            backToFullPack.putExtra("day", day);
            setResult(RESULT_OK, backToFullPack);
            finish();
            return true;
        }
        return super.onKeyDown(keyCode, event);
    }
```

```
    private class ReverseAdapter extends ArrayAdapter<String> {

        private ReverseAdapter(Context context) {
            super(context, android.R.layout.simple_list_item_activated_1,
packItems);
        }

        @Override
        public String getItem(int position) {
            return packItems.get(packItems.size() - position - 1);
        }
    }

    private class IconArrayAdapter extends ArrayAdapter<Integer> {
        private Integer[] images;

        private IconArrayAdapter(Context context, Integer[] images) {
            super(context, android.R.layout.select_dialog_item, images);
            this.images = images;
        }

        @Override
        public View getDropDownView(int position, View convertView,
ViewGroup parent) {
            return getImageForPosition(position);
        }

        @Override
        public View getView(int position, View convertView, ViewGroup
parent) {
            return getImageForPosition(position);
        }

        private View getImageForPosition(int position) {
            ImageView imageView = new ImageView(getContext());
            imageView.setBackgroundResource(images[position]);
            imageView.setLayoutParams(new
AbsListView.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT));
            return imageView;
        }
    }
}
```

2.5 content_start_screen.xml

В данном файле содержится информация о компонентах начального экрана и их взаимном расположении.

```
   <?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
```

```
    android:id="@+id/mainContainer"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.aleksandra.easybag.StartScreen"
    tools:showIn="@layout/start_screen">

    <TextView
        android:id="@+id/topHint"
        app:layout_constraintTop_toTopOf="parent"
        style="@style/TopHint"/>

    <ScrollView
        android:id="@+id/scrollContainer"
        android:layout_width="match_parent"
        android:layout_height="446dp"
        android:layout_marginTop="8dp"
        app:layout_constraintTop_toBottomOf="@+id/topHint">

        <LinearLayout
            android:id="@+id/horizontalLayout"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">


            <LinearLayout
                android:id="@+id/leftVerticalLayout"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center_horizontal"
                android:orientation="vertical">

            </LinearLayout>

            <LinearLayout
                android:id="@+id/rightVerticalLayout"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center_horizontal"
                android:orientation="vertical">

            </LinearLayout>
        </LinearLayout>

    </ScrollView>
</android.support.constraint.ConstraintLayout>
```

2.6 content_new_pack.xml

В данном файле содержится информация о компонентах окна создания нового списка и их взаимном расположении.

```
   <?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.aleksandra.easybag.NewPack"
    tools:showIn="@layout/new_pack">

    <ListView
        android:id="@+id/newPack"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/newItemLayout" />

    <Spinner
        android:id="@+id/spinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/placeLayout" />

    <android.support.design.widget.TextInputLayout
        android:id="@+id/nameLayout"
        style="@style/InputLayout"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:hint="@string/nameHint"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/nameNewPack"
            style="@style/AppTheme.EditText" />
    </android.support.design.widget.TextInputLayout>

    <android.support.design.widget.TextInputLayout
        android:id="@+id/placeLayout"
        style="@style/InputLayout"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:hint="@string/placeHint"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/nameLayout">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/newPlace"
            style="@style/AppTheme.EditText" />
    </android.support.design.widget.TextInputLayout>

    <android.support.design.widget.TextInputLayout
        android:id="@+id/newItemLayout"
        style="@style/InputLayout"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:hint="@string/newItem"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/editDateLayout">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/newItem"
            style="@style/AppTheme.EditText" />
    </android.support.design.widget.TextInputLayout>

    <android.support.design.widget.TextInputLayout
        android:id="@+id/editDateLayout"
        style="@style/InputLayoutDate"
        android:layout_width="180dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:hint="@string/dateHint"
        app:layout_constraintEnd_toStartOf="@+id/spinner"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/placeLayout">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/date"
            style="@style/AppTheme.EditTextDate" />
    </android.support.design.widget.TextInputLayout>

</android.support.constraint.ConstraintLayout>
```

2.7 content_full_pack.xml

В данном файле содержится информация о компонентах окна просмотра списка и их взаимном расположении.

```
   <?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.aleksandra.easybag.FullPack"
    tools:showIn="@layout/full_pack">

    <ListView
        android:id="@+id/fullPack"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/dateInfo" />

    <TextView
        android:id="@+id/placeInfo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/dateInfo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/placeInfo" />
</android.support.constraint.ConstraintLayout>
```

2.8 content_edit_pack.xml

В данном файле содержится информация о компонентах окна редактирования списка и их взаимном расположении.

```
   <?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mainContainer"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.aleksandra.easybag.EditPack"
    tools:showIn="@layout/edit_pack"
    android:descendantFocusability="beforeDescendants"
```

```
android:focusableInTouchMode="true">

<android.support.design.widget.TextInputLayout
    android:id="@+id/editNameLayout"
    style="@style/InputLayout"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:hint="@string/nameHint"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <android.support.design.widget.TextInputEditText
        android:id="@+id/editNamePack"
        style="@style/AppTheme.EditText" />
</android.support.design.widget.TextInputLayout>

<android.support.design.widget.TextInputLayout
    android:id="@+id/editPlaceLayout"
    style="@style/InputLayout"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:hint="@string/placeHint"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editNameLayout">

    <android.support.design.widget.TextInputEditText
        android:id="@+id/editPlace"
        style="@style/AppTheme.EditText" />
</android.support.design.widget.TextInputLayout>

<android.support.design.widget.TextInputLayout
    android:id="@+id/editItemLayout"
    style="@style/InputLayout"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:hint="@string/editNewItem"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editDateLayout">

    <android.support.design.widget.TextInputEditText
        android:id="@+id/editNewItem"
        style="@style/AppTheme.EditText" />
</android.support.design.widget.TextInputLayout>

<android.support.design.widget.TextInputLayout
    android:id="@+id/editDateLayout"
    style="@style/InputLayoutDate"
    android:layout_width="180dp"
```

```
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:hint="@string/dateHint"
        app:layout_constraintEnd_toStartOf="@+id/editSpinner"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/editPlaceLayout">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/editDate"
            style="@style/AppTheme.EditTextDate" />
    </android.support.design.widget.TextInputLayout>

    <Spinner
        android:id="@+id/editSpinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/editPlaceLayout" />

    <ListView
        android:id="@+id/editPack"
        style="@style/AppTheme.EditText"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/editItemLayout" />

</android.support.constraint.ConstraintLayout>
```

2.9 styles.xml

В данном файле содержится описания ститей компонентов мобильного приложения.

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
        <item name="android:textColor">@color/colorPrimary</item>
        <item name="android:typeface">serif</item>
        <item name="android:textSize">18sp</item>
    </style>
```

```
    <style name="AppTheme.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
        <item name="windowActionModeOverlay">true</item>
    </style>

    <style name="namedField">
        <item name="android:textSize">18sp</item>
        <item name="android:fontFamily">@font/micross</item>
    </style>

    <style name="AppTheme.AppBarOverlay"
parent="ThemeOverlay.AppCompat.Dark.ActionBar" />

    <style name="AppTheme.PopupOverlay"
parent="ThemeOverlay.AppCompat.Light" />

    <style name="DateButton">
        <item
name="android:background">@drawable/rectangle_rounded_some</item>
        <item name="android:padding">8dp</item>
        <item name="android:textAllCaps">false</item>
        <item name="android:textColor">#ffffff</item>
        <item name="android:textSize">18sp</item>
    </style>

    <style name="AppTheme.EditText">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:inputType">text</item>
        <item name="android:maxLines">1</item>
        <item name="android:textColorHint">#bcaaa4</item>
    </style>
    <style name="AppTheme.EditTextDate">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:inputType">date</item>
        <item name="android:maxLines">1</item>
        <item name="android:textColorHint">#bcaaa4</item>
    </style>

    <style name="InputLayout">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColorHint">#bcaaa4</item>
    </style>

    <style name="InputLayoutDate">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColorHint">#bcaaa4</item>
        <item name="android:focusable">false</item>
        <item name="android:editable">false</item>
    </style>
```

```
    <style name="TopHint">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textSize">14sp</item>
        <item name="android:textStyle">italic</item>
        <item name="android:paddingRight">8dp</item>
        <item name="android:paddingLeft">8dp</item>
        <item name="android:paddingTop">8dp</item>
        <item name="android:textAlignment">center</item>
        <item name="android:text">@string/top_hint</item>
    </style>

    <style name="PackName">
        <item name="android:textSize">18sp</item>
        <item name="android:textAlignment">center</item>
    </style>

</resources>
```

2.10 strings.xml

Данный файл содержит описания строковых констант, используемых в проекте.

```
   <resources>
    <string name="app_name">EasyBag</string>
    <string name="action_settings">Settings</string>
    <string name="top_hint">Создайте новую сумку (иконка +).</string>
    <string name="plus">Plus</string>
    <string name="edit">Edit</string>
    <string name="title_activity_new_pack">NewPack</string>
    <string name="newItem">Что берем с собой?</string>
    <string name="ic_save">Save</string>
    <string name="ic_delete">Delete</string>
    <string name="deleteItems">DeleteItems</string>
    <string name="title_activity_full_pack">FullPack</string>
    <string name="title_activity_edit_pack">EditPack</string>
    <string name="addNewItem">+ добавить новый пункт</string>
    <string name="nameHint">Название сумки</string>
    <string name="dateHint">Дата поездки</string>
    <string name="placeHint">Место назначения</string>
    <string name="iconHint">Изображение: </string>
    <string name="editNewItem">Новый пункт</string>
    <string name="alertDialogTitle">Удаление сумки</string>
    <string name="alertDialogMessage">Вы действительно хотите удалить
выбранную сумку?</string>
    <string name="positiveAnswer">Да</string>
    <string name="negativeAnswer">Отмена</string>
</resources>
```

2.11 colors.xml

Данный файл содержит описания цветов, используемых в проекте.

```
   <?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#ab2222</color>
    <color name="colorPrimaryDark">#932020</color>
    <color name="colorAccent">#8d6e63</color>
    <color name="accent_material_light_1">#009688</color>
    <color name="lightRed">#ef9a9a</color>
    <color name="white">#ffffff</color>
    <color name="lightBrown">#d7ccc8</color>
    <color name="background">#FFFAFAFA</color>
</resources>
```

2.12 start_screen_context.xml

Данный файл описание контекстного меню начального экрана.

```
   <?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/ic_delete"
        android:icon="@drawable/ic_delete_white"
        android:title="@string/ic_delete"
        app:showAsAction="always"/>
</menu>
```

2.13 new_pack_context.xml

Данный файл описание контекстного меню окна создания нового списка.

```
   <?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/deleteNewItems"
        android:icon="@drawable/ic_delete_white"
        android:orderInCategory="1"
        android:title="@string/deleteItems"
        app:showAsAction="always" />
</menu>
```

2.14 edit_pack_context.xml

Данный файл описание контекстного меню окна редактирования.

```
   <?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/deleteItems"
        android:icon="@drawable/ic_delete_white"
        android:orderInCategory="1"
        android:title="@string/deleteItems"
        app:showAsAction="always" />

</menu>
```

2.15 start_screen.xml

Данный файл описание верхнего меню начального экрана.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:app="http://schemas.android.com/apk/res-auto"
 xmlns:tools="http://schemas.android.com/tools"
 tools:context="com.example.aleksandra.easybag.StartScreen">
 <item
     android:id="@+id/plus"
     android:icon="@drawable/ic_plus_white"
     android:orderInCategory="1"
     android:title="@string/plus"
     app:showAsAction="always" />
</menu>
```

2.16 new_pack.xml

Данный файл описание верхнего меню окна создания нового списка.

```
<?xml version="1.0" encoding="utf-8"?>
<menu
 xmlns:app="http://schemas.android.com/apk/res-auto"
 xmlns:android="http://schemas.android.com/apk/res/android">
 <item
     android:id="@+id/ic_save"
     android:icon="@drawable/ic_check_white"
     android:title="@string/ic_save"
     app:showAsAction="always" />
</menu>
```

2.17 full_pack.xml

Данный файл описание верхнего меню окна просмотра списка.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:app="http://schemas.android.com/apk/res-auto">
 <item
     android:id="@+id/check"
     android:icon="@drawable/checkbox_white"
     android:orderInCategory="1"
     android:title="Checked"
     app:showAsAction="always" />

 <item
     android:id="@+id/edit"
     android:icon="@drawable/ic_pencil_white"
     android:orderInCategory="1"
     android:title="@string/edit"
     app:showAsAction="always" />

</menu>
```