ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК
КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

СИСТЕМА НАВИГАЦИИ В ЗДАНИИ «ГОСТЕХНАДЗОРА БЕЛГОРОДСКОЙ ОБЛАСТИ»

Выпускная квалификационная работа
обучающегося по направлению подготовки 02.03.03
«Математическое обеспечение и администрирование информационных систем»
очной формы обучения, группы 07001302
Данькова Николая Алексеевича

Научный руководитель:
доц., Чашин Ю. Г.

БЕЛГОРОД 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные навигационно-картографические системы имеют много различных форм и представлений, начиная с онлайн сервисов, доступных на компьютере, и заканчивая мобильными устройствами, которые всегда можно взять с собой – это автомобильные навигаторы, мобильные и стационарные платформы, веб-приложения и другие.

Такие системы предлагают своим пользователям возможности по нахождению оптимального пути из точки А в точку В и выводят, преимущественно на экран, инструкции по перемещению, выполнение которых проведет пользователя по маршруту.

Решению проблем гео- и локальной навигации посвящено большое количество работ, однако проблемы локальной навигации - внутри различных зданий и помещений остается актуальной. Арендованные на разных этажах кабинеты одной организации, здания с нестандартной планировкой, в которые вносятся планировочные изменения - не всегда оперативно учитываются. В сооружениях такого типа могут ориентироваться лишь сотрудники данных организаций. Таким образом, возникает потребность в сервисе, который поможет любому его пользователю просто и без траты лишнего времени добраться до нужного ему места в здании.

В качестве решения данной порблемы мы разберем пример навигации по зданию белгородского государственного надзора за техническим состоянием самоходных машин и других видов техники в Российской Федерации. Основной задачей государственного надзора за техническим состоянием самоходных машин и других видов техники в Российской Федерации (далее именуется как гостехнадзор) является осуществление надзора за техническим состоянием тракторов, самоходных дорожно-строительных и иных машин и прицепов к ним в процессе использования в части обеспечения безопасности для жизни, здоровья людей и имущества,

охраны окружающей среды, а в агропромышленном комплексе - за соблюдением правил эксплуатации машин и оборудования, регламентируемых стандартами, другими нормативными документами и документацией.[5]

Целью данной работы является разработка приложения для навигации по зданию «гостехнадзора Белгородской области».

Для достижения поставленной цели сформулированы задачи:

1. Анализ предметной области
2. Выбор алгоритма для реализации навигации
3. Выбор программных средств для реализации
4. Разработка приложения

Решая эти задачи мы должны получить систему, реализующую следующий функционал и отвечающую требованиям:

- функционал приложения заключается в построении оптимального маршрута по заданным данным и выводе его на карте;
- построение наиболее простых и понятных маршрутов;
- упрощение взаимодействия клиентов (посетителей) и зданий;
- один из ключевых параметров приложения это его доступность, поэтому принято решение о реализации Android-приложения;
- в связи с развития науки и постоянным нахождением новых методов и оптимизации старых, система должна быть спроектирована так, чтобы объем работ по изменению оптимизационного алгоритма был минимален.

Данная дипломная работа состоит из трех глав. В первой главе «Теоретические основы построения маршрутов» производится анализ существующих сервисов и выбор алгоритма построения маршрута. Во второй «Программная реализация системы навигации в здании «гостехнадзора Белгородской области» проводится выбор средств для реализации задачи. В третьей главе «Практическое применение программы» приведены результаты работы приложения.

1 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ МАРШРУТОВ

Первый раздел состоит из обзора предметной области, включающего в себя анализ уже существующих приложений, методов представления и построения маршрутов

1.1 Анализ существующих сервисов и их недостатки

Тема электронной картографии и навигации очень актуальна и востребована на сегодняшний день, этому способствовало стремительное развитие систем глобального позиционирования, телекоммуникационных технологий передачи данных на любые расстояния, информационных банков данных картографической информации.

В случае спутниковой навигации (GPS/Глонасс) существуют сервисы OutDoor, благодаря которым вы узнаете о ближайших кафе/ресторанах/гостиницах и т.д. благодаря тому, что известно ваше текущее местоположение. А благодаря сервисам indoor-навигации вы сможете без проблем и оперативно найти ближайшую стойку регистрации в здании аэропорта, экспонат в музее, полку с нужным вам товаром в магазине, кабинет с нужным отделом, и многое другое. Типичный пример – аудиогиды. Приходите в музей, берете аудиогид, и каждый раз вынуждены искать по номеру нужный экспонат, вводить его номер в устройство и слушать его описание. В случае применение indoor-навигации, всё производится автоматически – просто подойдите к заинтересовавшему вас экспонату, и его описание начнет проигрываться без дополнительных телодвижений с вашей стороны. Никаких сложностей и потери времени, всё просто.

Также, благодаря indoor-навигации появляются новые инструменты для маркетинга – проходя мимо вашего магазина, человек может моментально узнать о проводимых у вас акциях/мероприятиях/предоставляемых услугах, товарам, благодаря всплывающему сообщению на экране своего телефона

(так называемом “Geo-fencing”, причём предложенные ему предложения будут учитывать его интересы – т.к. можно учитывать информацию о его прошлых покупках), либо просто получить уведомление при приближении к определенному месту (второе направление indoor-навигации, называемое «Geo-aware»), а вы – получать статистическую информацию («тепловые карты» посетителей – своеобразный и очень мощный offline-аналог Google Analytics), основанные на перемещениях клиентов внутри ваших торговых залов (понять, какие отделы и товары пользуются повышенным интересом – очень легко). Рынок подобной геоконтекстной рекламы (LBA – location-based advertising) уже измеряется миллиардами долларов, и с развитием систем indoor-навигации ожидается его стремительный рост.

Такие системы, как: GPS, Galileo, ГЛОНАСС и др., обеспечивающие работу таких сервисов, как Google Maps, NAVIMIND, 2GIS и др., ориентированы на решение задач геонавигации и проблемы локальной навигации не решают.

«2ГИС» (произносится «двагúс»)— международная картографическая компания, выпускающая одноимённые электронные справочники с картами городов с 1999 года. [6]

Все версии «2ГИС», как и обновления к ним, бесплатны для пользователей. Основной источник доходов компании «2ГИС» — продажа рекламных мест на карте и в справочнике (баннер, место в списке, дополнительный текст).

Размеры современных мегаполисов увеличиваются с каждым днем, усложняя тем самым ориентацию в них городских жителей и гостей, которые посещают их с туристическими или рабочими визитами. Помочь с навигацией по улицам каменных джунглей сегодня без лишних проблем могут многочисленные картографические сервисы, однако они абсолютно бесполезны внутри огромных зданий общественного пользования. Разобраться в вертикальной и горизонтальной структуре последних поможет приложение Navimind.

Navimind — это современное решение для навигации по аэропортам, гипермаркетам, торговым центрам и другим масштабным зданиям, предназначенным для общественного пользования городских жителей и гостей мегаполисов. Оно содержит внутри себя огромное количество интерактивных схем подобных строений, которые позволят не только определиться с маршрутом следования к определенному месту, но и предоставят исчерпывающую информацию о специфике взаимодействия с ним.

Принцип работы Navimind очень прост – после загрузки приложения пользователь видит подробную поэтажную планировку интересующего здания. Приложение может определить местоположение пользователя на схеме либо ориентируясь на ближайший магазин, либо с помощью радиомаяков, если они установлены в здании.

После того как пользователь выбрал интересующий магазин, приложение практически мгновенно строит оптимальный маршрут к этому магазину от его текущего местоположения. Рассчитать наилучшие маршруты в здании, охватывающие необходимые вам магазины, теперь совсем просто. При этом вы значительно экономите драгоценное время, не заплутаете в переходах и коридорах, быстро совершая все запланированные покупки.

Не менее актуально приложение Navimind в аэропортах, где заблудиться довольно легко. С его помощью вы можете быть уверены, что не опоздаете на рейс, вовремя встретите прилетающих партнеров или родственников и найдете нужный выход на посадку. Сейчас в приложении присутствует аэропорт «Домодедово», в июле обещают добавить «Шереметьево» и «Внуково».

Помимо мобильного приложения Navimind предлагает удобные выносные схемы с планировками этажей на сайт комплекса, а также приложение для сенсорных киосков. Причем, данные для мобильного приложения, схемы на сайт и терминального приложения берутся из одного источника. Это очень удобно: вы внесли изменение один раз и почти

мгновенно данные обновились во всех приложениях. Но данный сервис реализован только для крупных ТЦ и аэропотртов.

Карты Google (англ. Google Maps) — набор приложений, построенных на основе бесплатного картографического сервиса и технологии, предоставляемых компанией Google. Созданы в 2005 году.

Сервис представляет собой карту и спутниковые снимки планеты Земля. Для многих регионов доступны высокодетализированные аэрофотоснимки (снятые с высоты 250—500 м), для некоторых — с возможностью просмотра под углом 45° с четырёх сторон света. Дополнительно предлагаются снимки Луны и Марса.

Чтобы воспользоваться сервисом поиска маршрутов, необходимо переключиться на вкладку «Проложить маршрут», после чего набрать адрес, откуда вы едете, в формате «Страна, Населенный пункт, Улица, Дом» (например, «Россия, Белгород, ул. Победы, 85») и адрес места назначения.

При построении маршрутов строится сразу несколько, а пользователю потом предоставляется выбор, где указано расстояние и время движения по каждому из них с учетом загруженности дорог. Все маршруты можно одновременно вывести на карту и сравнить их визуально. Разумеется, более одного маршрута будет построено только тогда, когда в этом есть смысл.

Сами маршруты вполне адекватны, и особых нареканий к ним нет. В момент построения можно задать три параметра, наиболее актуальным из которых является исключение платных дорог.

Процесс навигации вполне типичен. Приложение показывает расстояние до ближайшего маневра и его направление, своевременно предупреждает обо всем этом голосом. Сам маршрут на карте хорошо читается, а все маневры на нем отображаются белыми стрелками, что весьма удобно. Также на линии маршрута при его просмотре видны пробки.

При уходе с маршрута приложение использует логику «объезд» и не играет на нервах непрерывными просьбами развернуться при первой возможности.

Во-первых, нет отображения и предупреждения о камерах, а также об ограничениях скорости на текущем участке маршрута. И хотя абсолютное большинство водителей уже натренировалось не превышать скорость более чем на 20 км/ч, все равно случается задуматься или пропустить знак, и навигатор тут был бы неплохим помощником.

Во-вторых, нет возможности установки DPOI — по крайней мере, нам так и не удалось разобраться в том, как это можно сделать. Тем не менее, DPOI, установленные в приложении Waze, которое не так давно купила Google, отображаются и на Google Картах (не во всех регионах), подтверждение чему можно видеть на картах других стран.

Интерфейс приложения одновременно прост, красив и практичен. В некотором смысле, его также можно назвать спартанским: на экране минимум управляющих элементов и минимум деталей на карте. Контуры зданий появляются только на очень крупных масштабах, нет никакого нагромождения значков POI при прокрутке карты.

Таблица 1.1

Аналогичные сервисы

Название	Краткое описание	Достоинства	Недостатки
2GIS	2GIS – Российский картографический сервис, предоставляющий карты, справочники и информационное сопровождение.	Большое количество информации. Имеются приложения для всех платформ и устройств.	Отсутствуют карты зданий, местоположение определяется по GPS/ ГЛОНАСС.
NAVIMIND	NAVIMIND – навигационный сервис, предназначенный для навигации в торговых центрах	Наличие большого количества схем зданий. Система работает как на терминале, так и в мобильном приложении.	Имеется только приложение для IOS. Отсутствует автоматическое определение местоположения.
Google Maps	Google Maps –сервис, распространяемый компанией Google, представляющий собой графическую и спутниковую карты, а так же карту сети автодорог.	Большое количество карт различных городов, легкая встраиваемость в приложения.	Отсутствуют карты зданий. Местоположение определяется только по GPS.

Из приведенной таблицы 1.1 видно, что на данный момент на рынке навигационных сервисов нет системы, которая бы определяла местоположение устройства и эффективно прокладывала между двумя произвольными объектами внутри здания. Таким образом необходимость разработки навигационной системы, реализующей подобный функционал является актуальной на сегодняшний день. В связи с этим возникает проблема выбора средств визуализации карт зданий и методов построения оптимального маршрута.

1.2 Обзор методов представления маршрутов

Для представления маршрута идеально подходит граф. Первые сведения о графах как о схемах в виде наборов точек, соединенных между собой какими-либо линиями, появились в XVIII веке. Сначала эти сведения были разрозненными и относились главным образом к головоломкам, играм и развлечениям. Но в конце XIX века в связи с развитием топологии значительно возрос интерес к теории графов. В то время она рассматривалась как одна из глав топологии. Однако вскоре обнаружилось, что методы теории графов успешно могут применяться в различных областях дискретной математики, таких как программирование, теория логических схем и многотактных дискретных автоматов, теория бинарных отношений и т. д.

В общем случае граф – это множество V точек, определенным образом соединенных между собой линиями, необязательно прямыми. Точки множества V называются вершинами графа, а соединяющие их линии – ребрами. Вершины графа обычно нумеруют десятичными числами, но можно использовать и любые другие знаки. Если вершины пронумерованы, то ребра обозначают неупорядоченными парами номеров вершин. Каждую пару образуют номера тех вершин, которые соединены ребром.[2]

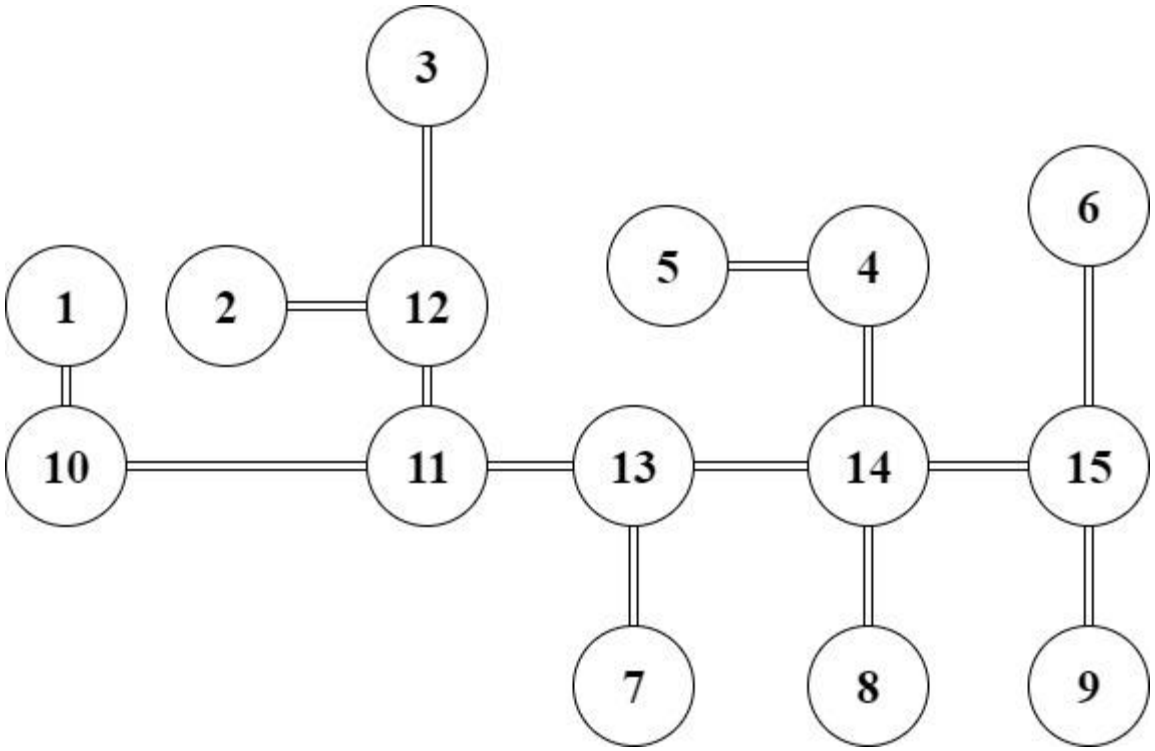

Рис.1.1. Граф

Для хранения и представления маршрутов в памяти электронного устройства рассмотрим несколько способов.

Матрица смежности – это еще один способ задания графов. Матрица смежности представляет собой квадратную таблицу размерами n × n, где n – число вершин графа.[2] Строкам и колонкам матрицы ставятся в соответствие вершины, а на пересечениях строк и колонок записываются числа, показывающие, сколько ребер соединяют соответствующие вершины графа. Построение матрицы смежности поясним на примере графа, приведенного на рис. 1.1. В графе пятнадцать вершин, следовательно, матрица смежности имеет пятнадцать строк и пятнадцать колонок. В первой строке слева записан нуль. Это значит, что вершина 1 не имеет петли. На десятой позиции от нуля записано число 1. Оно говорит о том, что вершины 1 и 10 соединены одним ребром и т. д. Матрица смежности показана в таблице 1.2.

Таблица 1.2

Матрица смежности

0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	1	1	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0	1	0	1
0	0	0	0	0	1	0	0	1	0	0	0	0	0	0

Матрица инцидентности — одно из представления графа, в котором отображаются связи между инцидентными элементами графа (ребра и вершины). Столбцы матрицы соответствуют ребрам, строки — вершинам. Значение, отличное от нуля, в ячейке матрицы указывает связь между вершиной и ребром (их инцидентность).[2]

В случае ориентированного графа каждой дуге (a,b) ставится в соответствующем столбце: «-1» в строке вершины a и «1» в строке вершины b; если связи между вершиной и ребром нет, то в соответствующую ячейку ставится «0».

Составим матрицу инцидентности, представленную в таблице 1.3, для графа с рис. 1.1.

Таблица 1.3

Матрица инцидентности

1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	0
0	1	1	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0	0	0	1	0	1	0
0	0	0	1	0	0	0	1	0	0	0	0	1	1
0	0	0	0	0	1	0	0	1	0	0	0	0	1

Исходя из поставленных задач, мы выберем матрицу инцидентности для хранения и представления графа в электронном виде.

1.3 Выбор метода построения маршрутов

Алгоритм Де́йкстры — алгоритм на графах, изобретённый нидерландским учёным Эдсгером Дейкстрой в 1959 году. Находит кратчайшие пути от одной из вершин графа до всех остальных. Алгоритм работает только для графов с рёбрами положительного веса. [2]

Определение: дан взвешенный ориентированный граф G(V,E) с дугами положительного веса. Найти кратчайшие пути от некоторой вершины a графа G до всех остальных вершин этого графа.

Е. Дейкстра разработал алгоритм для прохода по графам, дуги которых имеют разный вес. На каждом шаге, он ищет не пройденные узлы, близкие к начальному, далее просматривает соседей найденного узла, и устанавливает (обновляет) соответствующие расстояния от стартового узла. У этого

алгоритма несколько преимуществ по сравнению с поиском в ширину: он учитывает стоимость (длину) пути и обновляет узлы, если до них найден лучший путь.

Другой алгоритм Беллмана-Форда — алгоритм поиска кратчайшего пути во взвешенном графе. За время O(|V| × |E|) алгоритм находит кратчайшие пути от одной вершины графа до всех остальных. В отличие от алгоритма Дейкстры, алгоритм Беллмана-Форда допускает рёбра с отрицательным весом. Предложен независимо Ричардом Беллманом и Лестером Фордом.[2]

Определение: дан ориентированный (неориентированный) граф G со взвешенными рёбрами. Длиной пути назовём сумму весов рёбер, входящих в этот путь. Требуется найти кратчайшие пути от выделенной вершины s до всех вершин графа.

Заметим, что кратчайших путей может не существовать. Так, в графе, содержащем цикл с отрицательным суммарным весом, существует сколь угодно короткий путь от одной вершины этого цикла до другой (каждый обход цикла уменьшает длину пути). Цикл, сумма весов рёбер которого отрицательна, называется отрицательным циклом.

Лучший алгоритм для поиска оптимального пут в различных пространствах является A* ("A-звездочка"). Этот эвристический поиск перебирает все узлы по мере приближения к наилучшему маршруту идущему через этот узел.[2] Формула эвристики (1.1):

$$f(n) = g(n) + h(n) \quad (1.1)$$

где:

f(n) - оценка, назначенная узлу n;

g(n) наименьшая стоимость прибытия в узел n из начального;

h(n) эвристическая приближенная стоимость пути к цели от n-го узла.

Этот алгоритм учитывает длину предыдущего пути из алгоритма Дейкстры с эвристикой из алгоритма "лучший-первый". До тех пор пока

эвристическое приближение h(n) является допустимым, он однозначно находит кратчайший путь, то есть не превышает расстояния оставшегося до конечной цели. Этот алгоритм наилучшим образом использует эвристику: ни один алгоритм не пройдет меньшее число узлов, не учитывая при этом узлы с одинаковым весом.

В реальных задачах A* оказывается очень гибким. Состоянием зачастую является ячейка или позиция, которую занимает объект. В нашем случае это кабинет организации. Соседние состояния иногда могут изменяться по ситуации (напр. перепланировка здания). Смежные состояния могут быть исключены, потому что они непроходимы. Стоимость перехода из одной позиции в другую могут представлять разные вещи: обычное расстояние между позициями или стоимость по времени.

Для построения маршрута был выбран наиболее подходящий алгоритм A*, т.к. он отвечает требованиям в поставленных задачах.

2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ НАВИГАЦИИ В ЗДАНИИ «ГОСТЕХНАДЗОРА БЕЛГОРОДСКОЙ ОБЛАСТИ»

Второй раздел содержит описание средства реализации: выбор языка программирования и среды программирования. А также показано, почему был сделан выбор в пользу Java для разработки приложения на Android Studio.

2.1 Средства реализации

2.1.1 Выбор языка программирования

Выбор языка программирования основан на его возможностях и на том насколько они подходит для реализации поставленных задач

C++ — статически типизированный, компилируемый язык программирования.

Поддерживает такие методы программирования, как процедурное, объектно-ориентированное и обобщённое. Язык имеет обширную стандартную библиотеку, включающую в себя распространённые модули и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и др. C++ включает в себя свойства, принадлежащие как к высокоуровневым, так и низкоуровневым языкам. По сравнению с предшественником, языком C, в C++ наибольшее внимание уделено поддержке объектно-ориентированному программированию.

Этот язык является одним из самых популярных языков программирования. В область его применения входит создание разнообразных прикладных программ, драйверов устройств, операционных систем, приложений для встраиваемых систем, высокопроизводительных серверов, а также приложений. Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ.

Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. C++ оказал огромное влияние на другие языки программирования, в первую очередь на Java и C#.

Синтаксис C++ был наследован у языка C. Один из принципов разработки - сохранение совместимости с последним. Тем не менее, C++ не является в строгом смысле надмножеством C; программы, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно много, но не все они реализованы на C. Стандарт C++ состоит из двух частей:

- ядро языка, его описание
- описание стандартной библиотеки.

Сначала язык развивался спонтанно, не имея формальных рамок, по мере возникающих перед ним задач. Толчком к развитию языка послужило развитие кросс-компилятора cfront. Нововведения в языке отражались с изменением версии компилятора. Лишь в 1998 язык стал стандартизированным. C++ поддерживает:

- наследование;
- полиморфизм;
- инкапсуляция.

В C++ инкапсуляция реализуется с помощью указания уровня доступа к членам класса: они бывают открытыми (public), защищёнными (protected) и закрытыми (private). В C++ структуры отличаются от классов лишь формально - по умолчанию уровень доступы к членам класса и тип наследования у структуры публичные, а у класса — приватные.

С момента своего возникновения C++ подвергался серьезным ревизиям трижды, первый раз в 1985 году, второй — в 1989 году. Третий пересмотр языка произошел в связи с работой над стандартом ANSI для C++. Первая версия предложенного стандарта была создана к 25 января 1994 года. Комитет ANSI по языку C++ практически сохранил все черты языка, определенные Страуструпом, и добавил несколько новых. Процесс

стандартизации обычно является достаточно медленным, и стандартизация C++ не является исключением.

Изобретая C++ путем добавления к языку C поддержки объектно-ориентированного программирования, Страуструп представлял всю важность сохранения философии языка C, включая его эффективность, гибкость и то, что именно программист, а не язык отвечает за разрабатываемое программное обеспечение. Как будет видно, справиться с этой задачей было нелегко. C++ обеспечивает всю свободу языка C одновременно с мощью объектов. Как отмечал Страуструп, C++ позволяет добиться ясности, расширяемости и легкости сопровождения за счет структуризации причем без потери эффективности.

Хотя первоначально C++ был нацелен на работу с очень большими программами, это не ограничивает его применение. Фактически объектно-ориентированные атрибуты языка C++ могут быть эффективно применены фактически к любой задаче программирования. Этот язык часто используется для таких проектов, как создание редакторов, баз данных, персональных систем работы с файлами и коммуникационных программ. Благодаря тому, что C++ унаследовал эффективность языка C, с его помощью разрабатывается высокопроизводительное программное обеспечение.

Другой язык программирования – Java. Это сильно типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (после, приобретённой компанией Oracle). Приложения Java транслируются в специальный байт-код, следовательно, они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java-машины. [3]

Достоинством подобного способа исполнения программ является полная независимость байт-кода от ОС и оборудования, что позволяет выполнять Java-приложения на любых устройствах, для которых существует соответствующая виртуальная машина. Другая важная особенность Java - гибкая система безопасности, в рамках которой исполнение программы

контролируется виртуальной машиной. Любые операции, превышающие установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером), вызывают немедленное прерывание.

Часто к недостаткам использования виртуальной машины относят снижение производительности. Ряд усовершенствований несколько увеличил скорость выполнения программ на Java:

- применение технологии трансляции байт-кода в машинный код прямо во время работы программы (JIT-технология) с возможностью сохранения версий класса в машинном коде;
- использование платформоориентированного кода (native-код) в стандартных библиотеках;
- аппаратные средства, которые обеспечивают ускоренную обработку байт-кода.

Несмотря на то, что C++ в Android проектах можно использовать, основным языком все-таки остается Java. А C++ можно использовать для оптимизации сложных участков кода или при портировании существующих библиотек, которые переписывать на Java слишком трудоемко.

Технология Java протестирована, усовершенствована, расширена и проверена участниками сообщества разработчиков Java, архитекторов и энтузиастов. Java позволяет разрабатывать высокопроизводительные портативные приложения практически на всех компьютерных платформах. Доступность приложений в разнородных средах позволяет компаниям предоставлять более широкий спектр услуг, способствует повышению производительности, уровня взаимодействия и совместной работы конечных пользователей и существенному снижению стоимости совместного владения корпоративными и потребительскими приложениями. Java стала незаменимым инструментом для разработчиков и открыла для них следующие возможности:

- написание программного обеспечения на одной платформе и его запуск практически на любой другой платформе;
- создание программ, работающих в веб-браузере и имеющих доступ к веб-службам;
- разработка приложений на стороне сервера для форумов в Интернете, магазинов, опросов, обработки форм HTML и много другого;
- объединение приложений или служб с использованием языка Java для создания высокоспециализированных приложений или служб;
- создание многофункциональных и эффективных приложений для мобильных телефонов, удаленных процессоров, микроконтроллеров, беспроводных модулей, датчиков, шлюзов, потребительских продуктов и практически любых других категорий электронных устройств. [7]

Для программной реализации был выбран язык программирования Java, т.к. он больше всего подходит для создания Android приложений.

2.1.2 Выбор среды программирования

Выбор среды программирования основан на поставленных задач. Поскольку приложение для навигации должно быть удобным и всегда находится под рукой, решено было осуществить его реализацию на платформе Android.

Приложения под ОС Android являются программами в нестандартном байт-коде для виртуальной машины Dalvik, для них был разработан формат установочных пакетов .APK. Для работы с приложениями доступно множество библиотек: OpenGL (движок трёхмерной графики); Bionic; WebKit (готовый движок для веб-браузера; обрабатывает HTML, JavaScript); FreeType (движок обработки шрифтов). По сравнению с обычными приложениями Linux приложения Android подчиняются дополнительным правилам: Content Providers — обмен данными между приложениями; Notification Manager — доступ к строке состояния; Resource Manager —

доступ к таким ресурсам, как файлы XML, PNG, JPEG; Activity Manager — управление активными приложениями.[4]

Разработку приложений для Android можно вести на языке Java. Существует плагин для IntelliJ IDEA, облегчающий разработку Android-приложений, и для среды разработки NetBeans IDE. Кроме того, существует Motodev Studio for Android — комплексная среда разработки на базе Eclipse, позволяющая работать непосредственно с Google SDK.

В 2009 году в дополнение к ADT был опубликован Android Native Development Kit (NDK) — пакет инструментариев и библиотек, котрый позволяет реализовать часть, или полностью законченное, приложения на языке C/C++. NDK рекомендуется использовать при разработки приложения с участками кода, в которых необходима скорости выполнения команд.

В 2013-м году состоялся релиз Embarcadero RAD Studio — XE5. Возможность разработки нативных приложений для платформы Android. Процесс создания Android приложения не требует дополнительных устройств, кроме, собственно, Android устройства (в принципе, можно обойтись и эмулятором).

Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Наиболее известные приложения на основе Eclipse Platform — различные «Eclipse IDE» для разработки ПО на множестве языков

Eclipse служит, в основном, платформой для разработки расширений, чем он и популярен: любой разработчик может дополнить Eclipse своими модулями. Уже существуют различные модули, разрабатываемые инженерами QNX совместно с IBM, и средства для языков Ada (GNATbench, Hibachi), COBOL, FORTRAN, PHP, X10 (X10DT) и пр. от различных разработчиков. Множество расширений дополняет среду Eclipse диспетчерами для работы с БД, серверами приложений и др.

Eclipse JDT (Java Development Tools) — наиболее известный модуль, направленный на групповую разработку: среда интегрирована с системами управления версиями — CVS, GIT в основной поставке, для других систем существуют плагины. Также предлагает поддержку связи между IDE и системой управления задачами (ошибками). В силу доступности, Eclipse во многих организациях является корпоративным стандартом для разработки приложений.

Android Studio — это интегрированная среда разработки (IDE) для работы с платформой Android.

IDE находилась в свободном доступе начиная, опубликованная в мае 2013, а затем перешла в стадию бета-тестирования, начиная с версии 0.8, которая была выпущена в июне 2014 года. Первая стабильная версия 1.0 была выпущена в декабре 2014 года, тогда же прекратилась поддержка плагина Android Development Tools для Eclipse. [4]

Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains, официальное средство разработки Android приложений. Данная среда разработки доступна для Windows, OS X и Linux. 17 мая 2017 на ежегодной конференции Google I/O, Google анонсировал язык Kotlin используемый в Android Studio официальным языком программирования для платформы Android в добавление к Java и C++.

Для реализации поставленной задачи мы воспользуемся средой разработки Android Studio

2.2 Программная реализация

Исходя из поставленных задач были выбраны язык программирования – Java, и среда разработки – Android Studio. Разработка приложения будет основана на выбранных языке и среде программирования. На рисунке 2.1 показана структура приложения

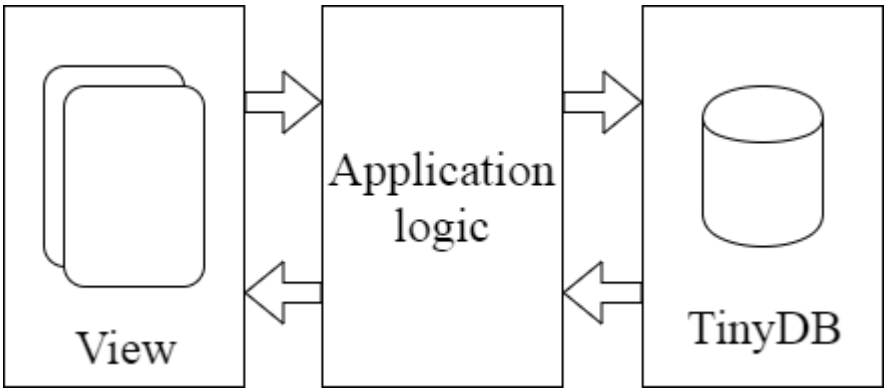

Рис 2.1. Структура приложения

В качестве базы данных была выбран TinyDB. Она представляет собой систему обработки запросов для извлечения информации из сети датчиков TinyOS.

В отличие от существующих решений для обработки данных в TinyOS, TinyDB не требует, написания кода C для датчиков. Вместо этого TinyDB обеспечивает простой SQL-подобный интерфейс, чтобы указать данные, которые мы хотим извлечь, наряду с дополнительными параметрами, как и скорость, с которой данные должны обновляться – по сравнению с запросами к традиционной базе данных. На рис. 2.2 представлены таблицы, используемые для хранения информации о кабинетах и сотрудниках.

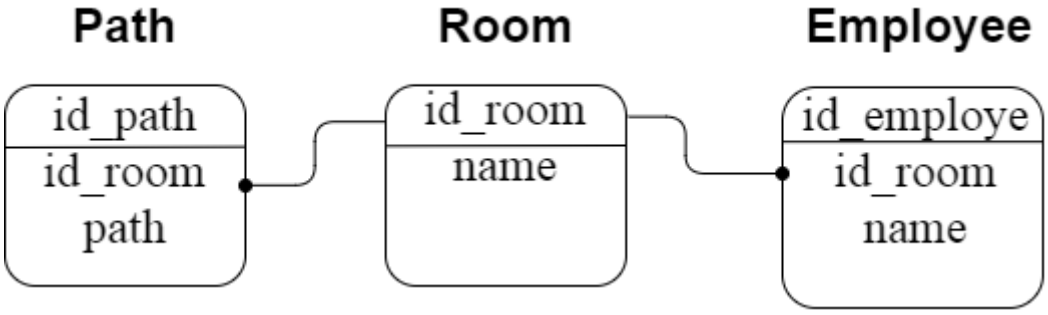

Рис. 2.2. Таблицы с данными о кабинетах и сотрудниках

На главном экране присутствуют два поля типа Spinner (выпадающий список). Выбрав значения и нажав кнопку «Поиск», вызывается метод Button.Click для передачи информации на другой экран (см. рис 2.3).

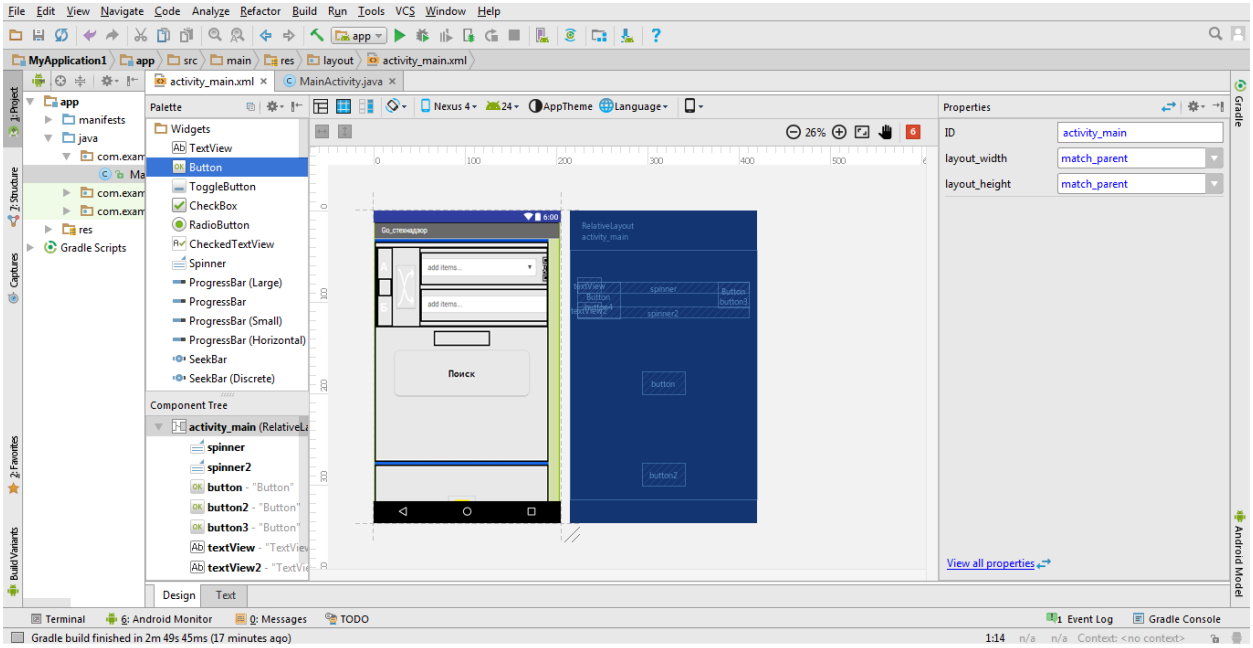

Рис.2.3. Основные элементы главного экрана

Также на рис.2.3 расположена кнопка для возможности считывания QR-кодов и использовать их значения в качестве пункта отправления (примеры QR-кодов представлены во вложении 2). Для этого использован компонент BarcodeScanner, функция обработки отсканированных значений представлена на листинге 2.1.

Листинг 2.1

```
.method public AfterScan(Ljava/lang/String;)V
    .locals 3
    .param p1, "result"    # Ljava/lang/String;
    .annotation runtime Lcom/google/app/components/annotations/SimpleEvent;
    .end annotation
    .prologue
    .line 128
    const-string v0, "AfterScan"
    const/4 v1, 0x1
    new-array v1, v1, [Ljava/lang/Object;
    const/4 v2, 0x0
    aput-object p1, v1, v2
    invoke-static {p0, v0, v1}, Lcom/google/app/components/runtime/EventDispatcher;-
>dispatchEvent(Lcom/google/app/components/runtime/Component;Ljava/lang/String;[Ljava/lan
g/Object;)Z
    .line 129
    return-void
 .end method
```

Данный компонент позволяет пользователю сканировать 1-D или 2-D «графические штрих - коды» с камерой на Android устройствах. Программа превращает то, что отсканировано в исходные данные, которая представлена штрих - кодом. Это позволяет пользователю получать веб - адреса, географические координаты, небольшие куски текста, и многое другое, просто наводя камеру своего устройства на штрих - код. Таким образом , система на базе Android работает почти как обычный считыватель штрих - кода, за исключением того , что состоит только из стандартных аппаратных средств и программного обеспечения этого приложения.

Есть много различных типов штрих - кодов, которые поддерживаются этим приложением, в дополнение к первоначальному 1-D штрих - коду. Они обычно предоставляются на товары, такие как продукты питания, книги, одежду, DVD, и большинство других продуктов , продаваемых на коммерческой основе. Когда приложение Barcode Scanner сканирует 1-D штрих - коды на одном из этих типов элементов, он может автоматически выполнять поиск в Интернете для этого продукта. С помощью этой функциональности, приложение может позволить пользователю быстро сделать сравнение между ценой в магазине с ценами из интернет - магазина.

Наиболее распространены 2-D штрих - коды, которые могут быть прочитаны с помощью этого приложения: QR – код и матрица данных штрих -кода. QR - коды часто встроены в веб - сайты так что, когда пользователи просматривают веб - сайт на своем настольном компьютере или ноутбуке и хотят передать ссылку, продукт или приложение на своё Android устройство, они могут быстро навести свою камеру на экран и с помощью сканера штрих -кодов, получить эти данные с веб - сайта.

Для использования быстрой смены пункта отправления/назначения присутствует swap-кнопка, функция которой приведена на листинге 2.2.

Листинг 2.2

```
public void Swap ( ){
        int a;
        a=Spinner1.IndexSelection;
        Spinner1.IndexSelection=Spinner2.IndexSelection;
        Spinner2.IndexSelection=a;
}
```

При выборе значений с SelectionIndex=0 (Spinner1.SelectionIndex, Spinner2.SelectionIndex) выводится сообщение с помощью Notifier (см. рис 2.4). Функция представлена в листинге 2.3.

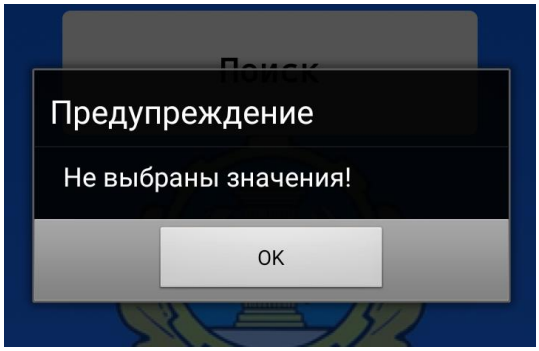

Рис.2.4. Компонент Notifier

Листинг 2.3

```
public void Errmess( ) {
   Notifier.showMessageDialog( this, 'Не выбраны значения!', 'Предупреждение', Notifier.DEFAULT_OPTION );
   // здесь ссылка this указывает на экземпляр класса MyMainFrame  }
```

После нажатия на кнопку «Поиск» открывается второй экран, обработка данной функции представлена в листинге 2.4, на который передается информация для построения маршрута и о пункте назначения (ФИО сотрудников).Пример второго экрана представлен на рис. 2.5.

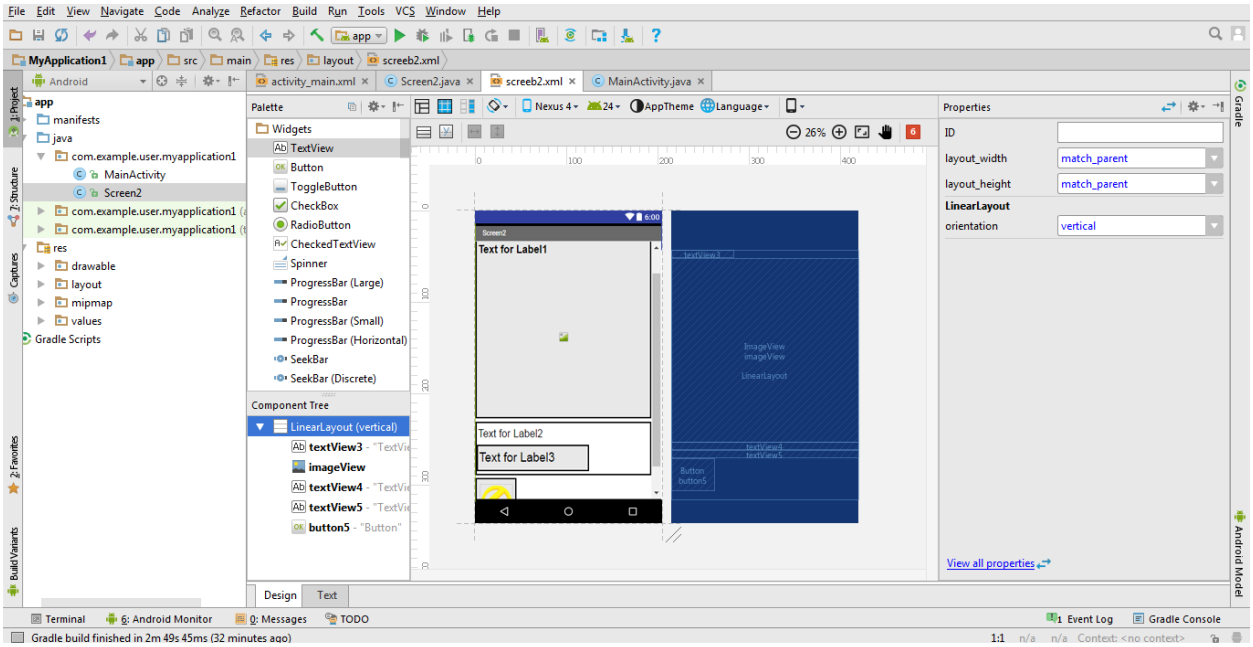

Рис. 2.5. Второй экран

Листинг 2.4

```
public void onClick(View view) {
ArrayList<String> List1 = new ArrayList<String>();
        // добавим в список ряд элементов
        List1.add(1,Spinner1.Selection);
        List1.add(2,Spinner2.Selection);
        List1.add(3,Spinner1.IndexSelection);
        List1.add(4,Spinner2.IndexSelection);
    EditText List1 = (List) findViewById(R.id.List1);
    Intent intent = new Intent(MainActivity.this, SecondActivity.class);
    intent.putExtra("list1", List1);
    startActivity(intent);
}
```

Для поиска маршрута используется алгоритм, выбранный в п. 1.3. В листинге 2.5 показан псевдокод алгоритма A*.

Листинг 2.5

```
приоритетная очередь Open
список Closed
ПоискАЗвездочка
  s.g = 0  // s - стартовый узел
  s.h = ЭвристическаяОценка( s )
  s.f = s.g + s.h
  s.родитель = null
  добавить s в Open
  пока очередь Open не пуста
    извлечь n из Open  // n - узел с наименьшей стоимость в Open
    если n целевой узел
      сконструировать путь
      выйти с кодом "успешное завершение"
    для каждого наследника n' узла n
      newg = n.g + стоимость(n,n')
      если n' в Open или Closed, и n'.g <= newg
   пропустить n'
      n'.родитель = n
      n'.g = newg
      n'.h = ЭвристическаяОценка( n' )
      n'.f = n'.g + n'.h
      если n' в Closed
        удалить n' из Closed
      если n' не в Open
        положить n' в Open
    положить n в Closed
  выйти с кодом "путь не найден"
```

В листинге 2.6 показана функция вывода ФИО сотрудников.

Листинг 2.6

```
public void GetFIO() {
        lable3.Text=FIO.Getvalue(findValueById(R.id.List1.get(4)))
}
```

Для вывода информации о сотрудниках выполняется поиск соответствия Spinner2.SelectionIndex и индексом в хранилище.

3 ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ПРОГРАММЫ

В данном разделе будет описана работа приложения: его функционал и компоненты, а также проведено тестирование.

3.1 Тестирование навигации в здании «гостехнадзора Белгородской области»

Открыв приложение, мы увидим его главный экран, рис 3.1. Он представляет собой два выпадающих списка, для выбора места отправления и назначения и кнопку «Поиск».

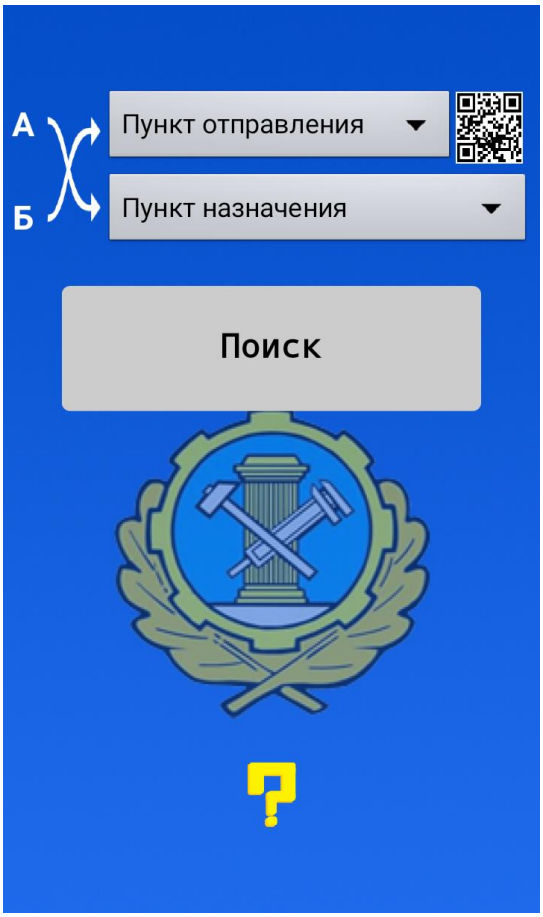

Рис.3.1. Основной экран

Также на нем присутствует:

- кнопка для сканирования QR-кодов, которая позволяет сканировать коды и использовать их значения для выбора как места отправления
- swap-кнопка для быстрой смены пунктов отправления и назначения между собой
- кнопка информации о приложении

Далее приведено подробное описание работы приложения.

Чтобы начать работу с приложением и найти маршрут, нужно выбрать пункты отправления назначения (рис. 3.2).

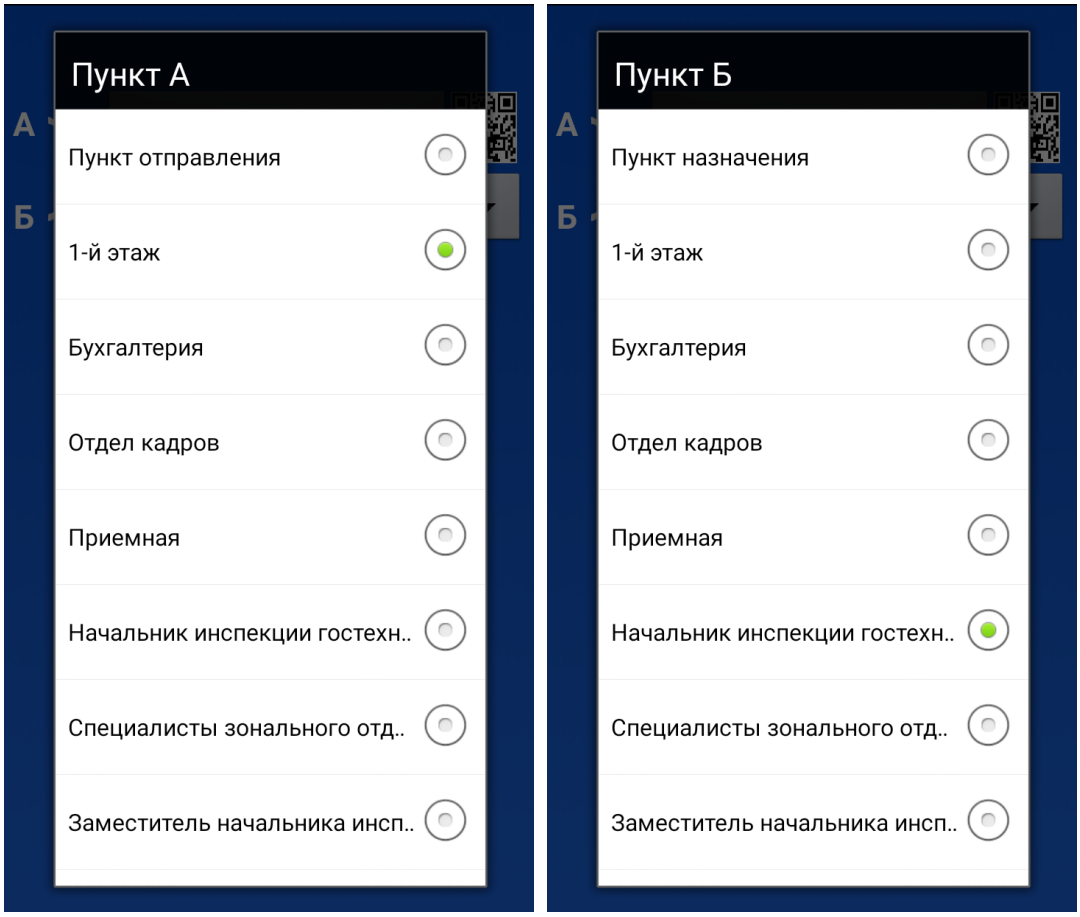

Рис.3.2. Выбор пункта отправления и назначения

Существуют всплывающие окна с информацией. На рис.3.3 представлено окно с информацией, сообщающей о том, что не выбраны значения.

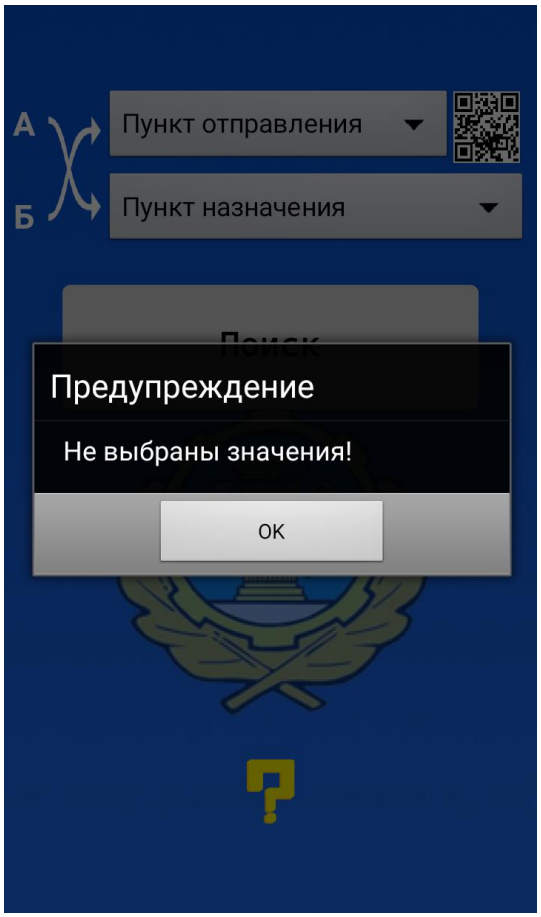

Рис.3.3. Всплывающее окно

При выборе одинаковых пунктов предусмотрен вывод специального сообщения (рис 3.4).

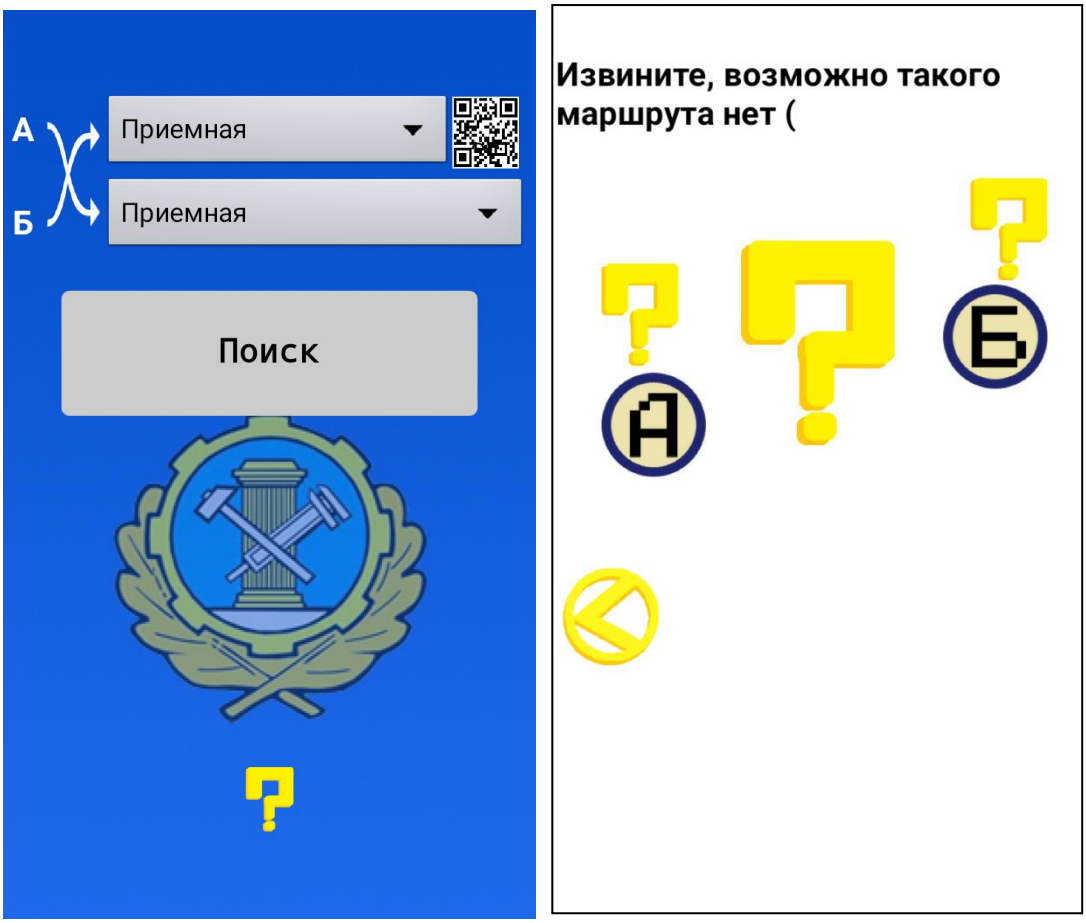

Рис.3.4. Выбор одинаковых пунктов и вывод сообщения

Выбрав нужные пункты и нажав кнопку «Поиск» загрузится новый экран, показанный на рис 3.5. На данном экране расположен маршрут, перечень сотрудников, находящихся в пункте назначения и кнопка для возврата на предыдущий экран.

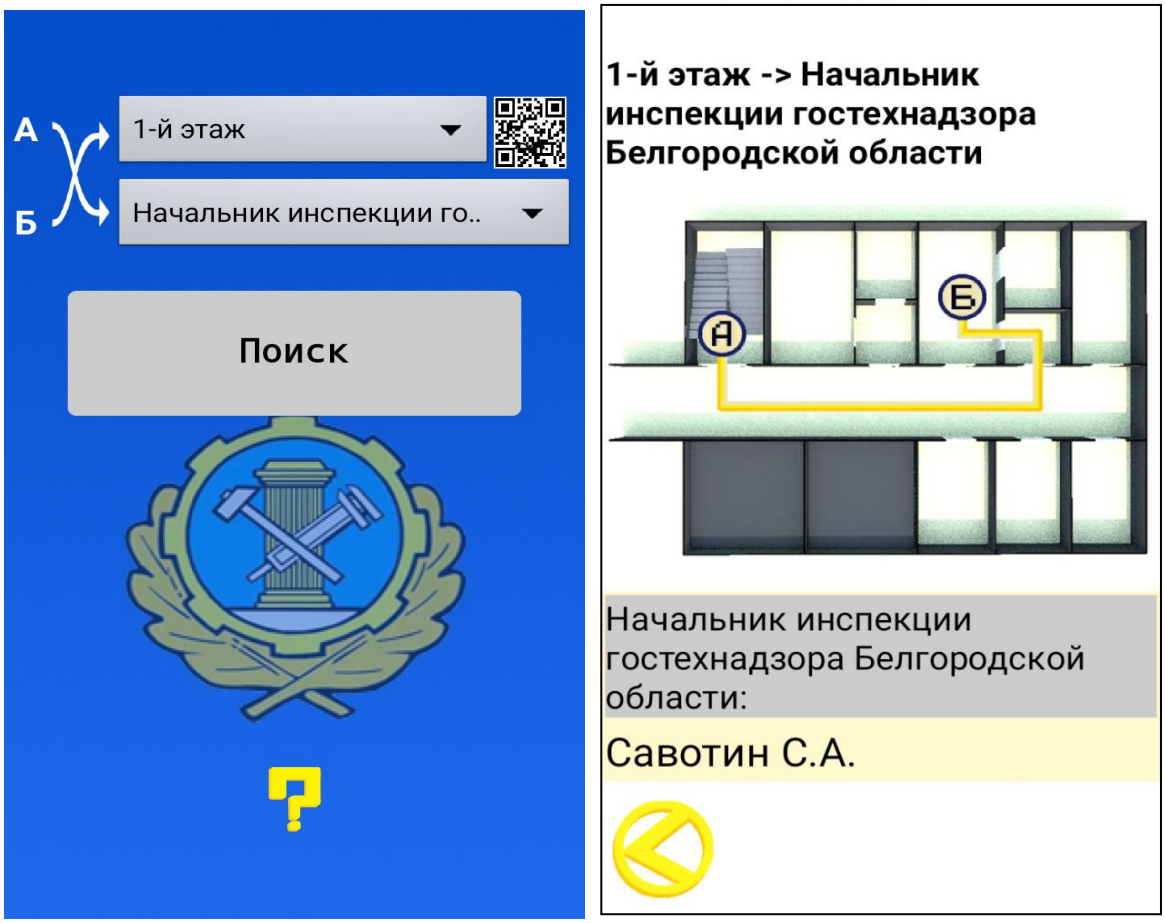

Рис.3.5. Выбор пунктов отправления/ назначения и результат поиска

Для использования быстрой смены пункты отправления/назначения можно нажать swap-кнопку (рис. 3.6). Далее показан пример работы данной кнопки.

Рис.3.6. Выбор пунктов отправления/ назначения и результат поиска

После нажатия swap-кнопки пункты назначения и отправления поменяются местами. Нажав «Поиск» можно увидеть обратный маршрут, представленный на рис. ниже

Рис.3.7. Смена пунктов отправления/ назначения по нажатию на swap-кнопку и результат поиска

При сканировании QR-кода, содержащего не подходящую информацию, также выводится предупреждение, указанное на рис. 3.13.

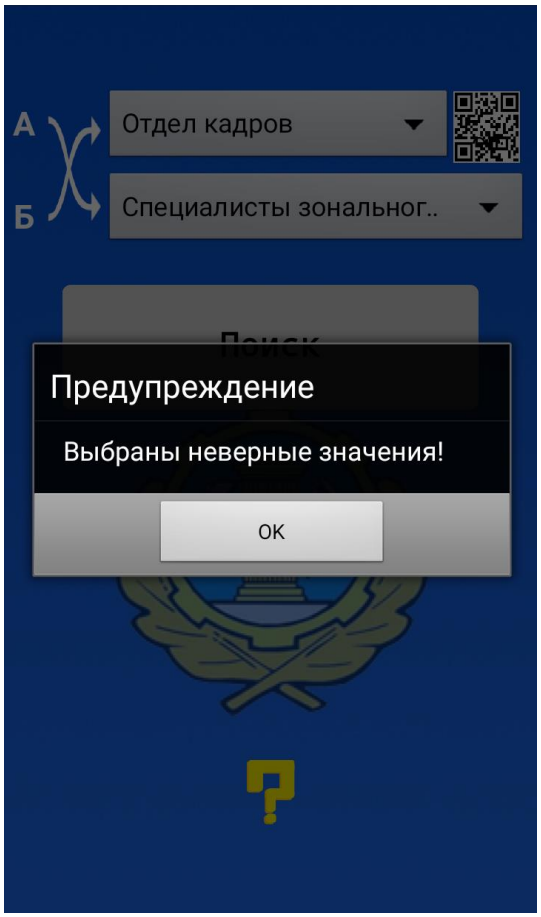

Рис.3.8. Предупреждение после сканирования неверного QR-кода

По нажатию на кнопку информации (на главном экране - знак вопроса), появляется информация о приложении (рис. 3.14).

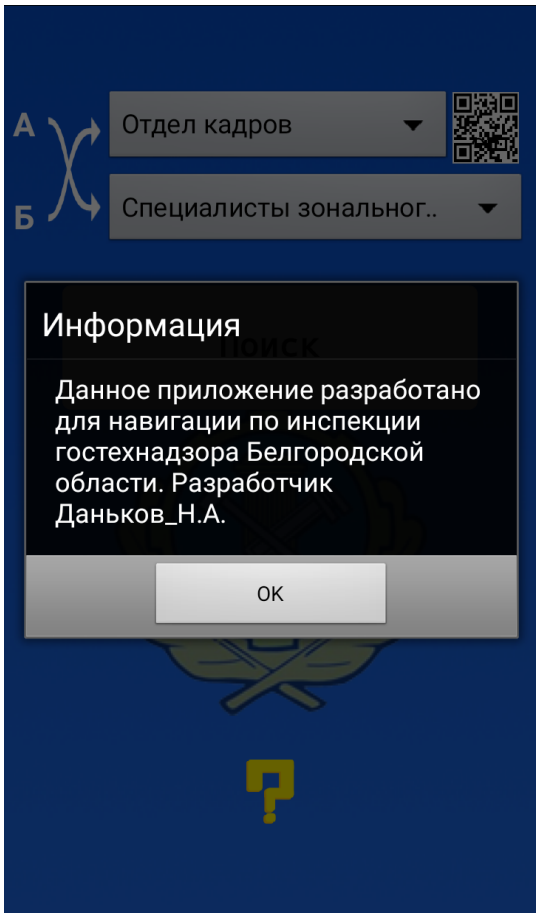

Рис.3.9. Информация о приложении

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной дипломной работы была достигнута поставленная цель, а именно: разработка приложения для навигации по зданию «гостехнадзора Белгородской области». Также реализованы поставленные задачи:

1. Анализ предметной области
2. Выбор алгоритма для реализации навигации
3. Выбор программных средств для реализации
4. Разработка приложения

В результате решения поставленных задач мы получили систему, реализующую функционал и отвечающую требованиям:

- функционал приложения заключается в построении оптимального маршрута по заданным данным и выводе его на карте;
- построение наиболее простых и понятных маршрутов;
- упрощение взаимодействия клиентов (посетителей) и зданий;
- один из ключевых параметров приложения это его доступность, поэтому оно реализовано в качестве Android-приложения;
- в связи с развития науки и постоянным нахождением новых методов и оптимизации старых, система спроектирована так, чтобы объем работ по изменению оптимизационного алгоритма был минимален.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Дубовик, Н.Н. Анализ структуры информационной системы для пространственной навигации / Дубовик Н.Н., Ногин О.А., Туманов В. М. Современные инновации №3(5)

2. Шевелев, Ю.П. Дискретная математика. Ч. 2: Теория конечных автоматов. Комбинаторика. Теория графов (для автоматизированной технологии обучения «Символ»): Учебное пособие./ Шевелев Ю.П. — Томск: Том. гос. ун-т систем упр. и радиоэлектроники, 2003. — 130 с.

3. Java [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Java (дата обращения 10.02.2017)

4. Android [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Android (дата обращения 11.02.2017)

5. Об инспекции гостехнадзора Белгородской области [Электронный ресурс] URL: http://www.белгтн.рф/about/ (дата обращения 04.04.2017)

6. 2ГИС [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/2ГИС (дата обращения 19.03.2017)

7. Подробнее о технологии Java [Электронный ресурс] URL: https://www.java.com/ru/about/ (дата обращения 17.03.2017)

Приложение 1

```
package app.ai_Dankov0001.new_app;
import com.google.app.components.common.PropertyTypeConstants;
import com.google.app.components.runtime.BarcodeScanner;
import com.google.app.components.runtime.Button;
import com.google.app.components.runtime.Component;
import com.google.app.components.runtime.EventDispatcher;
import com.google.app.components.runtime.Form;
import com.google.app.components.runtime.HandlesEventDispatching;
import com.google.app.components.runtime.HorizontalArrangement;
import com.google.app.components.runtime.Label;
import com.google.app.components.runtime.Notifier;
import com.google.app.components.runtime.Spinner;
import com.google.app.components.runtime.TinyDB;
import com.google.app.components.runtime.VerticalArrangement;
import com.google.app.components.runtime.errors.YailRuntimeError;
import com.google.app.components.runtime.util.RetValManager;
import com.google.app.components.runtime.util.RuntimeErrorAlert;

public class Screen1 extends Form implements Runnable {
    public static Screen1 Screen1;
    public Boolean $Stdebug$Mnform$St;
    public final ModuleMethod $define;
    public BarcodeScanner BarcodeScanner1;
    public final ModuleMethod BarcodeScanner1$AfterScan;
    public Button Button2;
    public final ModuleMethod Button2$Click;
    public Button Button3;
    public final ModuleMethod Button3$Click;
    public Button Button5;
    public final ModuleMethod Button5$Click;
    public Button Button6;
    public final ModuleMethod Button6$Click;
    public Label Label1;
    public Label Label3;
    public Label Label4;
    public Label Label5;
    public Notifier Notifier1;
    public final ModuleMethod Screen1$Initialize;
    public Spinner Spinner1;
    public Spinner Spinner2;
    public TinyDB TinyDB1;
    public final ModuleMethod add$Mnto$Mncomponents;
    public final ModuleMethod add$Mnto$Mnevents;
    public final ModuleMethod add$Mnto$Mnform$Mndo$Mnafter$Mncreation;
    public final ModuleMethod add$Mnto$Mnform$Mnenvironment;
    public final ModuleMethod add$Mnto$Mnglobal$Mnvar$Mnenvironment;
    public final ModuleMethod add$Mnto$Mnglobal$Mnvars;
    public final ModuleMethod android$Mnlog$Mnform;
    public LList components$Mnto$Mncreate;
    public final ModuleMethod dispatchEvent;
    public LList events$Mnto$Mnregister;
    public LList form$Mndo$Mnafter$Mncreation;
    public Environment form$Mnenvironment;
    public Symbol form$Mnname$Mnsymbol;
    public Environment global$Mnvar$Mnenvironment;
    public LList global$Mnvars$Mnto$Mncreate;
    public final ModuleMethod is$Mnbound$Mnin$Mnform$Mnenvironment;
    public final ModuleMethod lookup$Mnhandler;
    public final ModuleMethod lookup$Mnin$Mnform$Mnenvironment;
    public final ModuleMethod process$Mnexception;
```

```
    public final ModuleMethod send$Mnerror;
static {
        Lit175 = (SimpleSymbol) new SimpleSymbol("any").readResolve();
        Lit174 = (SimpleSymbol) new SimpleSymbol("lookup-handler").readResolve();
        Lit173 = (SimpleSymbol) new SimpleSymbol("dispatchEvent").readResolve();
        Lit172 = (SimpleSymbol) new SimpleSymbol("send-error").readResolve();
        Lit171 = (SimpleSymbol) new SimpleSymbol("add-to-form-do-after-creation").readResolve();
        Lit170 = (SimpleSymbol) new SimpleSymbol("add-to-global-vars").readResolve();
        Lit169 = (SimpleSymbol) new SimpleSymbol("add-to-components").readResolve();
        Lit168 = (SimpleSymbol) new SimpleSymbol("add-to-events").readResolve();
        Lit167 = (SimpleSymbol) new SimpleSymbol("add-to-global-var-environment").readResolve();
        Lit166 = (SimpleSymbol) new SimpleSymbol("is-bound-in-form-environment").readResolve();
        Lit165 = (SimpleSymbol) new SimpleSymbol("lookup-in-form-environment").readResolve();
        Lit164 = (SimpleSymbol) new SimpleSymbol("add-to-form-environment").readResolve();
        Lit163 = (SimpleSymbol) new SimpleSymbol("android-log-form").readResolve();
        Lit162 = new FString("com.google.app.components.runtime.Notifier");
        Lit161 = IntNum.make((int) Component.COLOR_NONE);
        Lit160 = IntNum.make((int) Component.COLOR_NONE);
        Lit159 = (SimpleSymbol) new SimpleSymbol("BackgroundColor").readResolve();
        Lit158 = new FString("com.google.app.components.runtime.Notifier");
        Lit157 = new FString("com.google.app.components.runtime.TinyDB");
        Lit156 = new FString("com.google.app.components.runtime.TinyDB");
        Lit155 = (SimpleSymbol) new SimpleSymbol("AfterScan").readResolve();
        Lit154 = (SimpleSymbol) new SimpleSymbol("BarcodeScanner1$AfterScan").readResolve();
        SimpleSymbol simpleSymbol = (SimpleSymbol) new
SimpleSymbol(PropertyTypeConstants.PROPERTY_TYPE_TEXT).readResolve();
        Lit15 = simpleSymbol;
     Lit153 = PairWithPosition.make(simpleSymbol, PairWithPosition.make(Lit15, PairWithPosition.make(Lit15,
LList.Empty, "/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1074224),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1074219),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1074213);
        simpleSymbol = (SimpleSymbol) new SimpleSymbol("number").readResolve();
        Lit11 = simpleSymbol;
        Lit152 = PairWithPosition.make(simpleSymbol, PairWithPosition.make(Lit11, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073867),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073859);
        Lit151 = PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073716),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073711);
        Lit150 = PairWithPosition.make(Lit11, PairWithPosition.make(Lit11, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073614),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073606);
        Lit149 = PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073480),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073475);
        Lit148 = (SimpleSymbol) new SimpleSymbol("$result").readResolve();
        Lit147 = PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073358),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073353);
```

```
        Lit146 = PairWithPosition.make(Lit11, PairWithPosition.make(Lit11, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073264),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 1073256);
        Lit145 = IntNum.make(10);
        Lit144 = new FString("com.google.app.components.runtime.BarcodeScanner");
        Lit143 = new FString("com.google.app.components.runtime.BarcodeScanner");
        Lit142 = new FString("com.google.app.components.runtime.Label");
        Lit141 = new FString("com.google.app.components.runtime.Label");
        Lit140 = new FString("com.google.app.components.runtime.Label");
        Lit139 = (SimpleSymbol) new SimpleSymbol("Visible").readResolve();
        Lit138 = (SimpleSymbol) new SimpleSymbol("HasMargins").readResolve();
        Lit137 = (SimpleSymbol) new SimpleSymbol("Label3").readResolve();
        Lit136 = new FString("com.google.app.components.runtime.Label");
        Lit135 = (SimpleSymbol) new SimpleSymbol("Button5$Click").readResolve();
        Lit134 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, PairWithPosition.make(Lit15,
LList.Empty, "/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 938862),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 938857),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 938851);
        Lit133 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 938764),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 938758);
        Lit132 = new FString("com.google.app.components.runtime.Button");
        Lit131 = (SimpleSymbol) new SimpleSymbol("Button5").readResolve();
        Lit130 = new FString("com.google.app.components.runtime.Button");
        Lit129 = new FString("com.google.app.components.runtime.HorizontalArrangement");
        Lit128 = IntNum.make(100);
        Lit127 = (SimpleSymbol) new SimpleSymbol("HorizontalArrangement4").readResolve();
        Lit126 = new FString("com.google.app.components.runtime.HorizontalArrangement");
        Lit125 = (SimpleSymbol) new SimpleSymbol("Button3$Click").readResolve();
        Lit124 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 833175),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 833169);
        Lit123 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 833039),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 833033);
        Lit122 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832915),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832909);
        Lit121 = (SimpleSymbol) new SimpleSymbol("StoreValue").readResolve();
        Lit120 = PairWithPosition.make((SimpleSymbol) new SimpleSymbol("list").readResolve(),
PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, PairWithPosition.make(Lit175,
PairWithPosition.make(Lit175, LList.Empty, "/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832770),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832766),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832762),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832758),
```

```
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832752);
      Lit119 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832734),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832728);
      Lit118 = PairWithPosition.make(Lit11, PairWithPosition.make(Lit11, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832709),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832701);
      Lit117 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832543),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832537);
      Lit116 = PairWithPosition.make(Lit11, PairWithPosition.make(Lit11, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832518),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832510);
      Lit115 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832352),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832346);
      Lit114 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832234),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832228);
      Lit113 = (SimpleSymbol) new SimpleSymbol("Selection").readResolve();
      Lit112 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit15, PairWithPosition.make(Lit15,
LList.Empty, "/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832033),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832028),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 832022);
      Lit111 = (SimpleSymbol) new SimpleSymbol("ShowMessageDialog").readResolve();
      Lit110 = (SimpleSymbol) new SimpleSymbol("Notifier1").readResolve();
      Lit109 = PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 831725),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 831720);
      Lit108 = PairWithPosition.make(Lit175, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 831612),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 831607);
      Lit107 = new FString("com.google.app.components.runtime.Button");
      Lit106 = (SimpleSymbol) new SimpleSymbol("Shape").readResolve();
      Lit105 = IntNum.make(250);
      Lit104 = IntNum.make(75);
      Lit103 = (SimpleSymbol) new SimpleSymbol("FontTypeface").readResolve();
      Lit102 = (SimpleSymbol) new SimpleSymbol("Button3").readResolve();
      Lit101 = new FString("com.google.app.components.runtime.Button");
      Lit100 = new FString("com.google.app.components.runtime.HorizontalArrangement");
      Lit99 = (SimpleSymbol) new SimpleSymbol("HorizontalArrangement1").readResolve();
      Lit98 = new FString("com.google.app.components.runtime.HorizontalArrangement");
      Lit97 = new FString("com.google.app.components.runtime.Spinner");
```

```
Lit96 = IntNum.make(255);
Lit95 = new FString("com.google.app.components.runtime.Spinner");
Lit94 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit93 = (SimpleSymbol) new SimpleSymbol("HorizontalArrangement2").readResolve();
Lit92 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit91 = (SimpleSymbol) new SimpleSymbol("Button2$Click").readResolve();
Lit90 = (SimpleSymbol) new SimpleSymbol("DoScan").readResolve();
Lit89 = (SimpleSymbol) new SimpleSymbol("BarcodeScanner1").readResolve();
Lit88 = new FString("com.google.app.components.runtime.Button");
Lit87 = IntNum.make(40);
Lit86 = IntNum.make(43);
Lit85 = (SimpleSymbol) new SimpleSymbol("Button2").readResolve();
Lit84 = new FString("com.google.app.components.runtime.Button");
Lit83 = new FString("com.google.app.components.runtime.Spinner");
Lit82 = (SimpleSymbol) new SimpleSymbol("Prompt").readResolve();
Lit81 = IntNum.make(210);
Lit80 = (SimpleSymbol) new SimpleSymbol("ElementsFromString").readResolve();
Lit79 = new FString("com.google.app.components.runtime.Spinner");
Lit78 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit77 = IntNum.make(300);
Lit76 = IntNum.make(50);
Lit75 = (SimpleSymbol) new SimpleSymbol("HorizontalArrangement3").readResolve();
Lit74 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit73 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit72 = (SimpleSymbol) new SimpleSymbol("VerticalArrangement3").readResolve();
Lit71 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit70 = (SimpleSymbol) new SimpleSymbol("Click").readResolve();
Lit69 = (SimpleSymbol) new SimpleSymbol("Button6$Click").readResolve();
Lit68 = (SimpleSymbol) new SimpleSymbol("Label4").readResolve();
Lit67 = new FString("com.google.app.components.runtime.Button");
Lit66 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
Lit65 = IntNum.make(35);
Lit64 = IntNum.make(70);
Lit63 = (SimpleSymbol) new SimpleSymbol("Button6").readResolve();
Lit62 = new FString("com.google.app.components.runtime.Button");
Lit61 = new FString("com.google.app.components.runtime.Label");
int[] iArr = new int[2];
iArr[0] = -1;
Lit60 = IntNum.make(iArr);
Lit59 = (SimpleSymbol) new SimpleSymbol("Label5").readResolve();
Lit58 = new FString("com.google.app.components.runtime.Label");
Lit57 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit56 = (SimpleSymbol) new SimpleSymbol("VerticalArrangement4").readResolve();
Lit55 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit54 = new FString("com.google.app.components.runtime.Label");
iArr = new int[2];
iArr[0] = -1;
Lit53 = IntNum.make(iArr);
Lit52 = (SimpleSymbol) new SimpleSymbol("TextColor").readResolve();
Lit51 = (SimpleSymbol) new SimpleSymbol("Text").readResolve();
Lit50 = IntNum.make(20);
Lit49 = (SimpleSymbol) new SimpleSymbol("FontSize").readResolve();
Lit48 = (SimpleSymbol) new SimpleSymbol("FontBold").readResolve();
Lit47 = (SimpleSymbol) new SimpleSymbol("Label1").readResolve();
Lit46 = new FString("com.google.app.components.runtime.Label");
Lit45 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit44 = IntNum.make(25);
Lit43 = (SimpleSymbol) new SimpleSymbol("VerticalArrangement2").readResolve();
Lit42 = new FString("com.google.app.components.runtime.VerticalArrangement");
Lit41 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit40 = (SimpleSymbol) new SimpleSymbol("HorizontalArrangement6").readResolve();
Lit39 = new FString("com.google.app.components.runtime.HorizontalArrangement");
Lit38 = new FString("com.google.app.components.runtime.VerticalArrangement");
```

```
        Lit37 = IntNum.make(-2);
        Lit36 = (SimpleSymbol) new SimpleSymbol("Width").readResolve();
        Lit35 = IntNum.make(350);
        Lit34 = (SimpleSymbol) new SimpleSymbol("Height").readResolve();
        Lit33 = (SimpleSymbol) new SimpleSymbol("VerticalArrangement1").readResolve();
        Lit32 = new FString("com.google.app.components.runtime.VerticalArrangement");
        Lit31 = (SimpleSymbol) new SimpleSymbol("Initialize").readResolve();
        Lit30 = (SimpleSymbol) new SimpleSymbol("Screen1$Initialize").readResolve();
        Lit29 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 106779),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 106773);
        Lit28 = (SimpleSymbol) new SimpleSymbol("Spinner2").readResolve();
        Lit27 = PairWithPosition.make(Lit15, PairWithPosition.make(Lit175, LList.Empty,
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 106633),
"/tmp/1496253616644_0.043716099597723046-
0/youngandroidproject/../src/app/ai_Dankov0001/new_app/Screen1.yail", 106627);
        Lit26 = (SimpleSymbol) new SimpleSymbol("GetValue").readResolve();
        Lit25 = (SimpleSymbol) new SimpleSymbol("TinyDB1").readResolve();
        Lit24 = (SimpleSymbol) new SimpleSymbol("SelectionIndex").readResolve();
        Lit23 = (SimpleSymbol) new SimpleSymbol("Spinner1").readResolve();
        Lit22 = (SimpleSymbol) new SimpleSymbol("VersionName").readResolve();
        Lit21 = (SimpleSymbol) new
SimpleSymbol(PropertyTypeConstants.PROPERTY_TYPE_BOOLEAN).readResolve();
        Lit20 = (SimpleSymbol) new SimpleSymbol("TitleVisible").readResolve();
        Lit19 = (SimpleSymbol) new SimpleSymbol("Title").readResolve();
        Lit18 = (SimpleSymbol) new SimpleSymbol("ScreenOrientation").readResolve();
        Lit17 = (SimpleSymbol) new SimpleSymbol("Icon").readResolve();
        Lit16 = (SimpleSymbol) new SimpleSymbol("BackgroundImage").readResolve();
        Lit14 = (SimpleSymbol) new SimpleSymbol("AppName").readResolve();
        Lit13 = IntNum.make(2);
        Lit12 = (SimpleSymbol) new SimpleSymbol("AlignVertical").readResolve();
        Lit10 = IntNum.make(3);
        Lit9 = (SimpleSymbol) new SimpleSymbol("AlignHorizontal").readResolve();
        Lit8 = (SimpleSymbol) new SimpleSymbol("g$list").readResolve();
        Lit7 = (SimpleSymbol) new SimpleSymbol("g$count").readResolve();
        Lit6 = IntNum.make(0);
        Lit5 = (SimpleSymbol) new SimpleSymbol("g$res").readResolve();
        Lit4 = IntNum.make(1);
        Lit3 = (SimpleSymbol) new SimpleSymbol("g$i").readResolve();
        Lit2 = (SimpleSymbol) new SimpleSymbol("*the-null-value*").readResolve();
        Lit1 = (SimpleSymbol) new SimpleSymbol("getMessage").readResolve();
        Lit0 = (SimpleSymbol) new SimpleSymbol("Screen1").readResolve();
    }
    public Screen1() {
        ModuleInfo.register(this);
        ModuleBody app_ai_Dankov0001_new_app_Screen1_frame = new frame();
        app_ai_Dankov0001_new_app_Screen1_frame.$main = this;
        this.android$Mnlog$Mnform = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 1, Lit163,
4097);
        this.add$Mnto$Mnform$Mnenvironment = new
ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 2, Lit164, 8194);
        this.lookup$Mnin$Mnform$Mnenvironment = new
ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 3, Lit165, 8193);
        this.is$Mnbound$Mnin$Mnform$Mnenvironment = new
ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 5, Lit166, 4097);
        this.add$Mnto$Mnglobal$Mnvar$Mnenvironment = new
ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 6, Lit167, 8194);
        this.add$Mnto$Mnevents = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 7, Lit168,
8194);
```

```
      this.add$Mnto$Mncomponents = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 8,
Lit169, 16388);
      this.add$Mnto$Mnglobal$Mnvars = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 9,
Lit170, 8194);
      this.add$Mnto$Mnform$Mndo$Mnafter$Mncreation = new
ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 10, Lit171, 4097);
      this.send$Mnerror = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 11, Lit172, 4097);
      this.process$Mnexception = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 12, "process-
exception", 4097);
      this.dispatchEvent = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 13, Lit173, 16388);
      this.lookup$Mnhandler = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 14, Lit174,
8194);
      PropertySet moduleMethod = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 15, null,
0);
      moduleMethod.setProperty("source-location", "/tmp/runtime6913560939214948 30.scm:552");
      lambda$Fn1 = moduleMethod;
      this.BarcodeScanner1$AfterScan = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 67,
Lit154, 4097);
      lambda$Fn47 = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 68, null, 0);
      lambda$Fn48 = new ModuleMethod(app_ai_Dankov0001_new_app_Screen1_frame, 69, null, 0);
   }
   public Object lookupInFormEnvironment(Symbol symbol) {
      return lookupInFormEnvironment(symbol, Boolean.FALSE);
   }

   public void run() {
      Throwable th;
      CallContext instance = CallContext.getInstance();
      Consumer consumer = instance.consumer;
      instance.consumer = VoidConsumer.instance;
      try {
         run(instance);
         th = null;
      } catch (Throwable th2) {
         th = th2;
      }
      ModuleBody.runCleanup(instance, th, consumer);
   }
public final void run(CallContext $ctx) {
      Consumer $result = $ctx.consumer;
      Object find = require.find("com.google.youngandroid.runtime");
      try {
         String str;
         ((Runnable) find).run();
         this.$Stdebug$Mnform$St = Boolean.FALSE;
         this.form$Mnenvironment = Environment.make(misc.symbol$To$String(Lit0));
         FString stringAppend = strings.stringAppend(misc.symbol$To$String(Lit0), "-global-vars");
         if (stringAppend == null) {
            str = null;
         } else {
            str = stringAppend.toString();
         }
         this.global$Mnvar$Mnenvironment = Environment.make(str);
         Screen1 = null;
         this.form$Mnname$Mnsymbol = Lit0;
         this.events$Mnto$Mnregister = LList.Empty;
         this.components$Mnto$Mncreate = LList.Empty;
         this.global$Mnvars$Mnto$Mncreate = LList.Empty;
         this.form$Mndo$Mnafter$Mncreation = LList.Empty;
         find = require.find("com.google.youngandroid.runtime");
         try {
            ((Runnable) find).run();
            if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
```

```
            Values.writeValues(runtime.addGlobalVarToCurrentFormEnvironment(Lit3, Lit4), $result);
          } else {
            addToGlobalVars(Lit3, lambda$Fn2);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            Values.writeValues(runtime.addGlobalVarToCurrentFormEnvironment(Lit5, Lit6), $result);
          } else {
            addToGlobalVars(Lit5, lambda$Fn3);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            Values.writeValues(runtime.addGlobalVarToCurrentFormEnvironment(Lit7, Lit6), $result);
          } else {
            addToGlobalVars(Lit7, lambda$Fn4);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            Values.writeValues(runtime.addGlobalVarToCurrentFormEnvironment(Lit8,
runtime.callYailPrimitive(runtime.make$Mnyail$Mnlist, LList.Empty, LList.Empty, "make a list")), $result);
          } else {
            addToGlobalVars(Lit8, lambda$Fn5);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            runtime.setAndCoerceProperty$Ex(Lit0, Lit9, Lit10, Lit11);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit12, Lit13, Lit11);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit14,
"Go_\u0441\u0442\u0435\u0445\u043d\u0430\u0434\u0437\u043e\u0440", Lit15);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit16, "log1_1.png", Lit15);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit17, "log2.png", Lit15);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit18, "portrait", Lit15);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit19,
"Go_\u0441\u0442\u0435\u0445\u043d\u0430\u0434\u0437\u043e\u0440", Lit15);
            runtime.setAndCoerceProperty$Ex(Lit0, Lit20, Boolean.FALSE, Lit21);
            Values.writeValues(runtime.setAndCoerceProperty$Ex(Lit0, Lit22, "1.1", Lit15), $result);
          } else {
            addToFormDoAfterCreation(new Promise(lambda$Fn6));
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            runtime.addToCurrentFormEnvironment(Lit30, this.Screen1$Initialize);
          } else {
            addToFormEnvironment(Lit30, this.Screen1$Initialize);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            EventDispatcher.registerEventForDelegation((HandlesEventDispatching) runtime.$Stthis$Mnform$St,
"Screen1", "Initialize");
          } else {
  if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            runtime.addToCurrentFormEnvironment(Lit69, this.Button6$Click);
          } else {
            addToFormEnvironment(Lit69, this.Button6$Click);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            EventDispatcher.registerEventForDelegation((HandlesEventDispatching) runtime.$Stthis$Mnform$St,
"Button6", "Click");
          } else {
if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            runtime.addToCurrentFormEnvironment(Lit91, this.Button2$Click);
          } else {
            addToFormEnvironment(Lit91, this.Button2$Click);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            EventDispatcher.registerEventForDelegation((HandlesEventDispatching) runtime.$Stthis$Mnform$St,
"Button2", "Click");
          } else {
  if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
```

```
            runtime.addToCurrentFormEnvironment(Lit125, this.Button3$Click);
          } else {
            addToFormEnvironment(Lit125, this.Button3$Click);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            EventDispatcher.registerEventForDelegation((HandlesEventDispatching) runtime.$Stthis$Mnform$St,
"Button3", "Click");
          } else {
  if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            runtime.addToCurrentFormEnvironment(Lit154, this.BarcodeScanner1$AfterScan);
          } else {
            addToFormEnvironment(Lit154, this.BarcodeScanner1$AfterScan);
          }
          if (runtime.$Stthis$Mnis$Mnthe$Mnrepl$St != Boolean.FALSE) {
            EventDispatcher.registerEventForDelegation((HandlesEventDispatching) runtime.$Stthis$Mnform$St,
"BarcodeScanner1", "AfterScan");
          } else {
  runtime.initRuntime();
        } catch (ClassCastException e) {
          throw new WrongType(e, "java.lang.Runnable.run()", 1, find);
        }
      } catch (ClassCastException e2) {
        throw new WrongType(e2, "java.lang.Runnable.run()", 1, find);
      }
   }
 public Object Button3$Click() {
      runtime.setThisForm();
      if (runtime.processOrDelayed$V(new Object[]{lambda$Fn35, lambda$Fn36}) != Boolean.FALSE) {
        return runtime.callComponentMethod(Lit110, Lit111, LList.list3, "OK"), Lit112);
      }
      ModuleMethod moduleMethod = runtime.yail$Mnlist$Mnadd$Mnto$Mnlist$Ex;
      Pair list1 = LList.list1(runtime.lookupGlobalVarInCurrentFormEnvironment(Lit8,
runtime.$Stthe$Mnnull$Mnvalue$St));
      LList.chain4(list1, runtime.callYailPrimitive(strings.string$Mnappend,
LList.list2(runtime.getProperty$1(Lit23, Lit113), ElementType.MATCH_ANY_LOCALNAME), Lit114, "join"),
runtime.callYailPrimitive(strings.string$Mnappend, LList.list2(runtime.getProperty$1(Lit28, Lit113),
ElementType.MATCH_ANY_LOCALNAME), Lit115, "join"), runtime.callYailPrimitive(strings.string$Mnappend,
LList.list2(runtime.callYailPrimitive(AddOp.$Mn, LList.list2(runtime.getProperty$1(Lit23, Lit24), Lit4), Lit116, "-
"), ElementType.MATCH_ANY_LOCALNAME), Lit117, "join"),
runtime.callYailPrimitive(strings.string$Mnappend, LList.list2(runtime.callYailPrimitive(AddOp.$Mn,
LList.list2(runtime.getProperty$1(Lit28, Lit24), Lit4), Lit118, "-"), ElementType.MATCH_ANY_LOCALNAME),
Lit119, "join"));
      runtime.callYailPrimitive(moduleMethod, list1, Lit120, "add items to list");
      runtime.callComponentMethod(Lit25, Lit121, LList.list2("s1", runtime.getProperty$1(Lit23, Lit24)), Lit122);
      runtime.callComponentMethod(Lit25, Lit121, LList.list2("s2", runtime.getProperty$1(Lit28, Lit24)), Lit123);
      runtime.addGlobalVarToCurrentFormEnvironment(Lit5, Lit4);
      return runtime.callYailPrimitive(runtime.open$Mnanother$Mnscreen$Mnwith$Mnstart$Mnvalue,
LList.list2("Screen2", runtime.lookupGlobalVarInCurrentFormEnvironment(Lit8,
runtime.$Stthe$Mnnull$Mnvalue$St)), Lit124, "open another screen with start value");
   }
 public Object BarcodeScanner1$AfterScan(Object $result) {
      $result = runtime.sanitizeComponentData($result);
      runtime.setThisForm();
      runtime.addGlobalVarToCurrentFormEnvironment(Lit3, Lit4);
      while (true) {
        if (runtime.processAndDelayed$V(new Object[]{lambda$Fn45, lambda$Fn46}) == Boolean.FALSE) {
          break;
        }
        Object signalRuntimeError;
        ModuleMethod moduleMethod = runtime.yail$Mnequal$Qu;
        if ($result instanceof Package) {
          signalRuntimeError = runtime.signalRuntimeError(strings.stringAppend("The variable ",
runtime.getDisplayRepresentation(Lit148), " is not bound in the current context"), "Unbound Variable");
```

```
        } else {
            signalRuntimeError = $result;
        }
        if (runtime.callYailPrimitive(moduleMethod, LList.list2(signalRuntimeError,
runtime.lookupGlobalVarInCurrentFormEnvironment(Lit3, runtime.$Stthe$Mnnull$Mnvalue$St)), Lit149, "=") !=
Boolean.FALSE) {
            runtime.addGlobalVarToCurrentFormEnvironment(Lit7, Lit4);
        } else {
            runtime.addGlobalVarToCurrentFormEnvironment(Lit3, runtime.callYailPrimitive(AddOp.$Pl,
LList.list2(runtime.lookupGlobalVarInCurrentFormEnvironment(Lit3, runtime.$Stthe$Mnnull$Mnvalue$St), Lit4),
Lit150, "+"));
        }
    }
    if (runtime.callYailPrimitive(runtime.yail$Mnequal$Qu,
LList.list2(runtime.lookupGlobalVarInCurrentFormEnvironment(Lit7, runtime.$Stthe$Mnnull$Mnvalue$St), Lit4),
Lit151, "=") == Boolean.FALSE) {
        return runtime.callComponentMethod(Lit110, Lit111, LList.list3, "OK"), Lit153);
    }
    SimpleSymbol simpleSymbol = Lit23;
    SimpleSymbol simpleSymbol2 = Lit24;
    AddOp addOp = AddOp.$Pl;
    if ($result instanceof Package) {
        $result = runtime.signalRuntimeError(strings.stringAppend("The variable ",
runtime.getDisplayRepresentation(Lit148), " is not bound in the current context"), "Unbound Variable");
    }
    return runtime.setAndCoerceProperty$Ex(simpleSymbol, simpleSymbol2, runtime.callYailPrimitive(addOp,
LList.list2($result, Lit4), Lit152, "+"), Lit11);
  }
```

Приложение 2

1-й этаж

Бухгалтерия

Отдел кадров

Приемная

Начальник инспекции гостехнадзора Белгородской области

Специалисты зонального отдела

Заместитель начальника инспекции гостехнадзора Белгородской области

Системные администраторы

Специалисты зонального отдела (консультанты)