**Министерство образования и науки**

Белгородский государственный технологический университет

им. В.Г. Шухова

Кафедра программного обеспечения вычислительной техники

и автоматизированных систем

**КУРСОВАЯ РАБОТА**

по дисциплине «Основы программирования»

тема: «Клиентское приложение на базе ОС Android для

транспортной информационной системы»

Автор работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Воскобойников И. С.

(подпись) ВТ-22

Руководитель проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Брусенцева В.С.

(подпись)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Белгород

2019 г.

Оглавление

[Введение 3](#_Toc26213039)

[Постановка задачи 3](#_Toc26213040)

[Обоснование выбора решения задачи 4](#_Toc26213041)

[Алгоритм в укрупнённых блоках 5](#_Toc26213042)

[Описание функций 6](#_Toc26213043)

[Тестовые данные 13](#_Toc26213044)

[Разбиение программы на модули 15](#_Toc26213045)

[Заключение 16](#_Toc26213046)

[Список используемой литературы: 16](#_Toc26213047)

[Приложения 16](#_Toc26213048)

# Введение

Смартфоны давно перестали быть чем- то необычным и стали обыденностью нашей жизни. При этом они обладают такой большой функциональностью, что практически стоят на ровне с компьютерами, а самое главное - они мобильны и всегда находятся под рукой. Каковы же причины распространения данной операционной системы?

Во-первых, Android поддерживает большое количество устройств разных производителей.

Во-вторых, Android характеризуется высокой доступностью средств разработки. Средства разработки для платформы Android бесплатны, в то время как разработка, к примеру, под iPhone (от компании Apple) требует немалых начальных финансовых вложений. Кроме всего вышеперечисленного, преимуществом ОС Android является наличие бесплатных библиотек для работы со сторонними ресурсами (Yandex MapKit, Google Map API, др.). Именно поэтому, приложение будет разрабатываться для смартфонов под управлением Android ОС.

Стрейбус - информационная система для пассажирских перевозок с динамической маршрутизацией.

Транспортная задача горожанина добраться из одного места города в другое за минимальное время, минимальные деньги и с приемлемыми удобствами. Стрейбус - тот проект, который позволит решить эту задачу.

Задача, которую требуется нам решить, связана с созданием удобного для клиента мобильного приложения. Мобильное приложение для пассажира позволяет оформить поездку, получить информацию об автобусе, его времени прибытия и времени в пути, а также оплатить поездку.

# Постановка задачи

Разработать алгоритм и создать модульную программу на языке Java, для пассажиров автобуса, позволяющую получить информацию об автобусе, его времени прибытия и времени в пути.

# Обоснование выбора решения задачи

Для решения задачи был выбран следующий вариант решения.

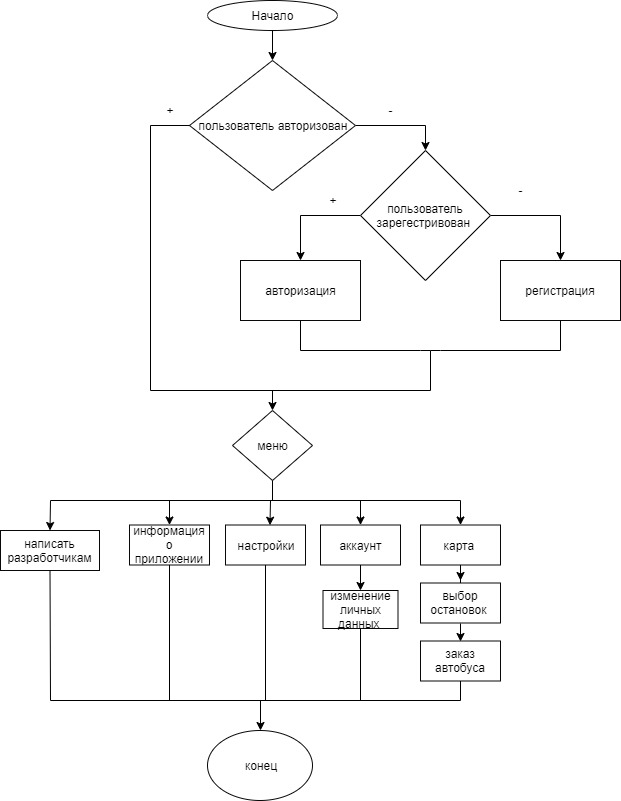
Пользователь, если входит в приложение первый раз, попадает на окно авторизации. Если у пользователя есть аккаунт, он авторизовывается, а если нет - может перейти на окно регистрации. После успешной авторизации пользователь попадает на карту с метками остановок.

Следующий шаг –это выбор начальной (место отправления) и конечной (место прибытия) остановок.

Из пунктов меню пользователь может попасть на карту, в настройки аккаунта, в настройки приложения, получить информацию о приложении, а так же, в случае необходимости, написать в тех поддержку .

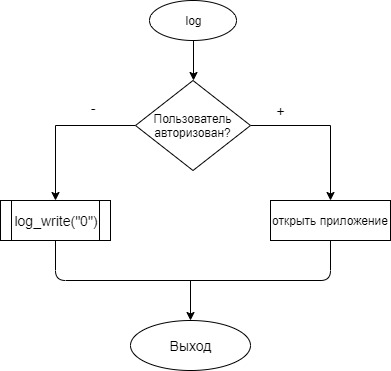
В настройках аккаунта можно сменить пароль, почту или сменить пользователя.

# Алгоритм в укрупнённых блоках



# Описание функций

1. Заголовок: void log ()
2. Назначение: Если авторизация была осуществлена ранее, то открыть само приложение иначе записать в файл отвечающий за вход “0”



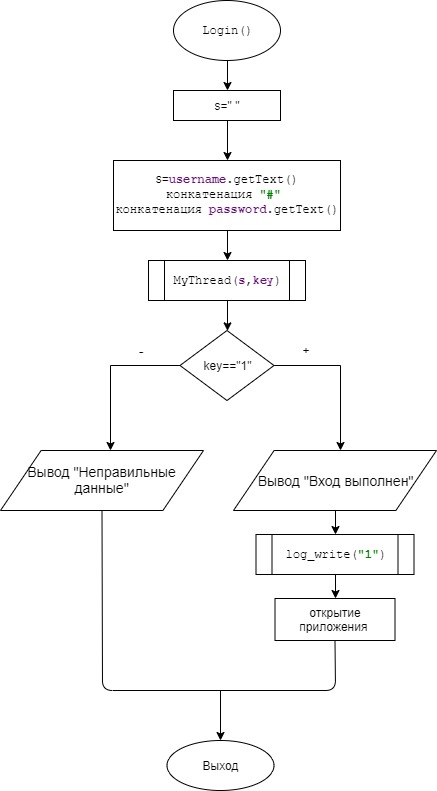
1. Заголовок: void log\_write (String t)
2. Назначение: записывает в файл fos содержание переменной t.



1. Заголовок: public void registration(View view)
2. Назначение: по нажатии на кнопку “Регистрация” открывает окно регистрации.

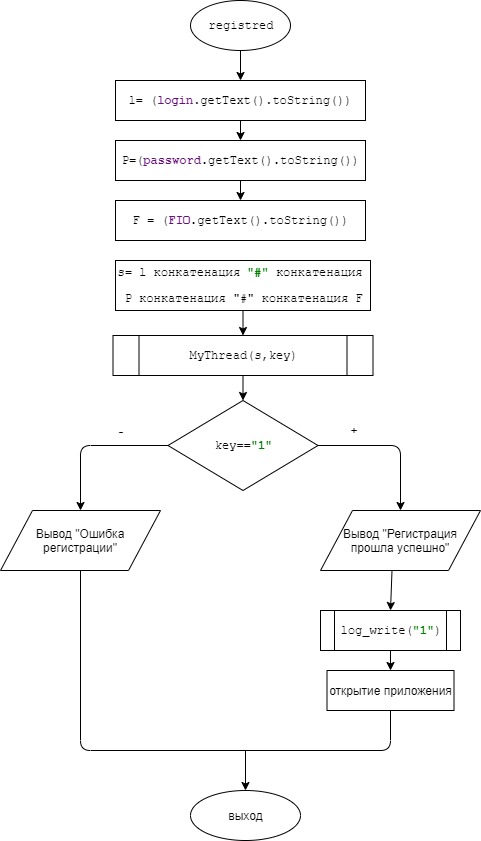
1. Заголовок: public void Login (View view)

2. Назначение: по нажатии на кнопку “Вход” отправляет введенные “логин” и” пароль” на сервер если получен ответ “1”, то выводит “Вход выполнен”, записывает в файл данные входа и открывает приложение.



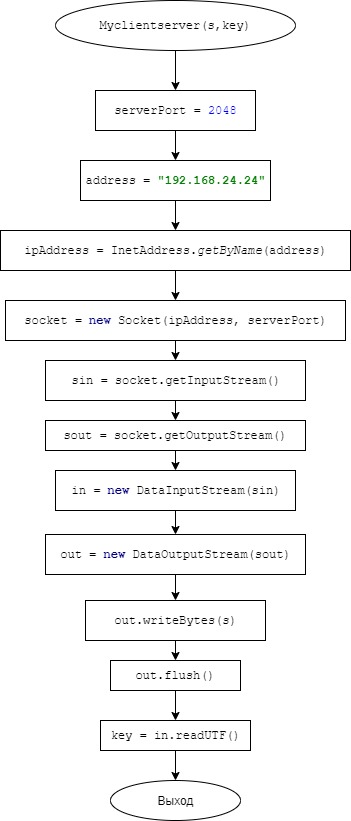
1. Заголовок: public void registred (View view)

2. Назначение: по нажатии на кнопку “Регистрация” отправляет введенные “ФИО”, “логин” и” пароль” на сервер если получен ответ “1”, то выводит “Вход выполнен”, записывает в файл данные входа и открывает приложение.



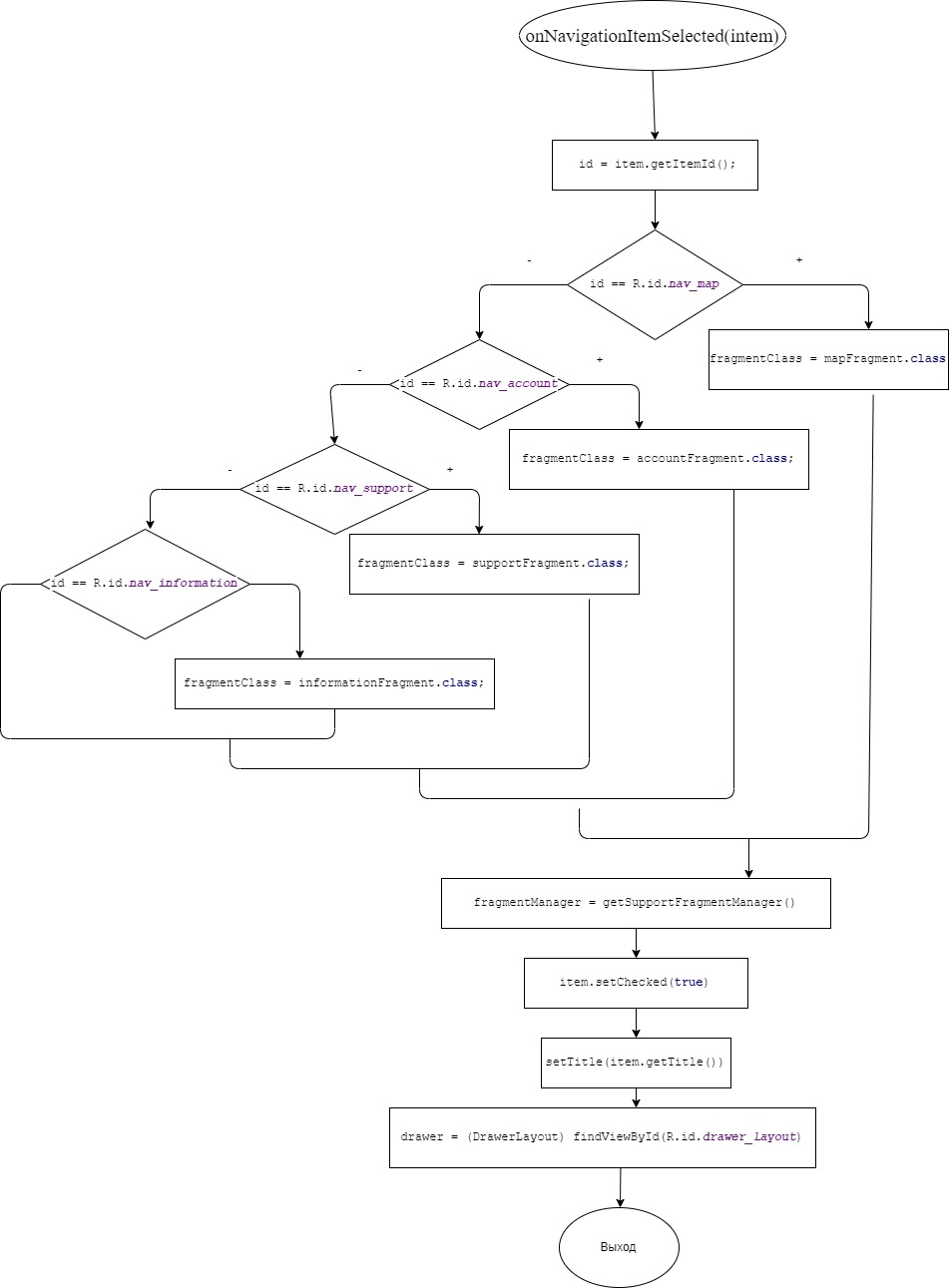
1. Заголовок: public Myclientserver(String s, String key)

2. Назначение: подключается к серверу и отправляет на него s и получает от него ответ key

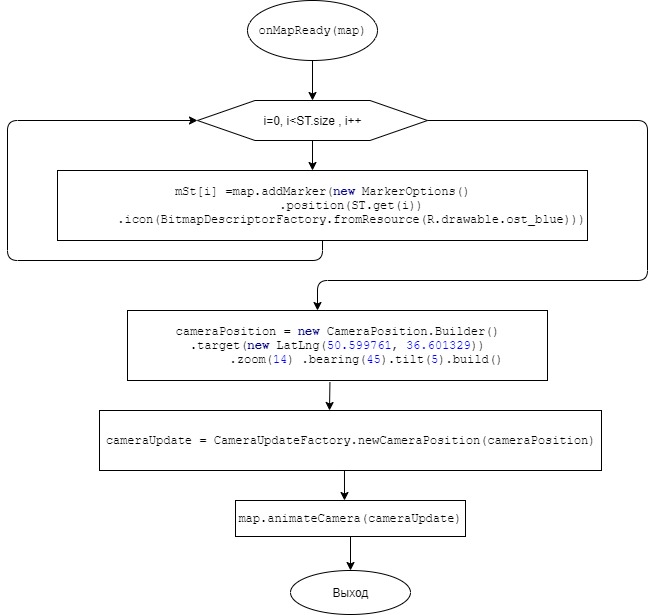


1. Заголовок: public boolean onNavigationItemSelected(MenuItem item)

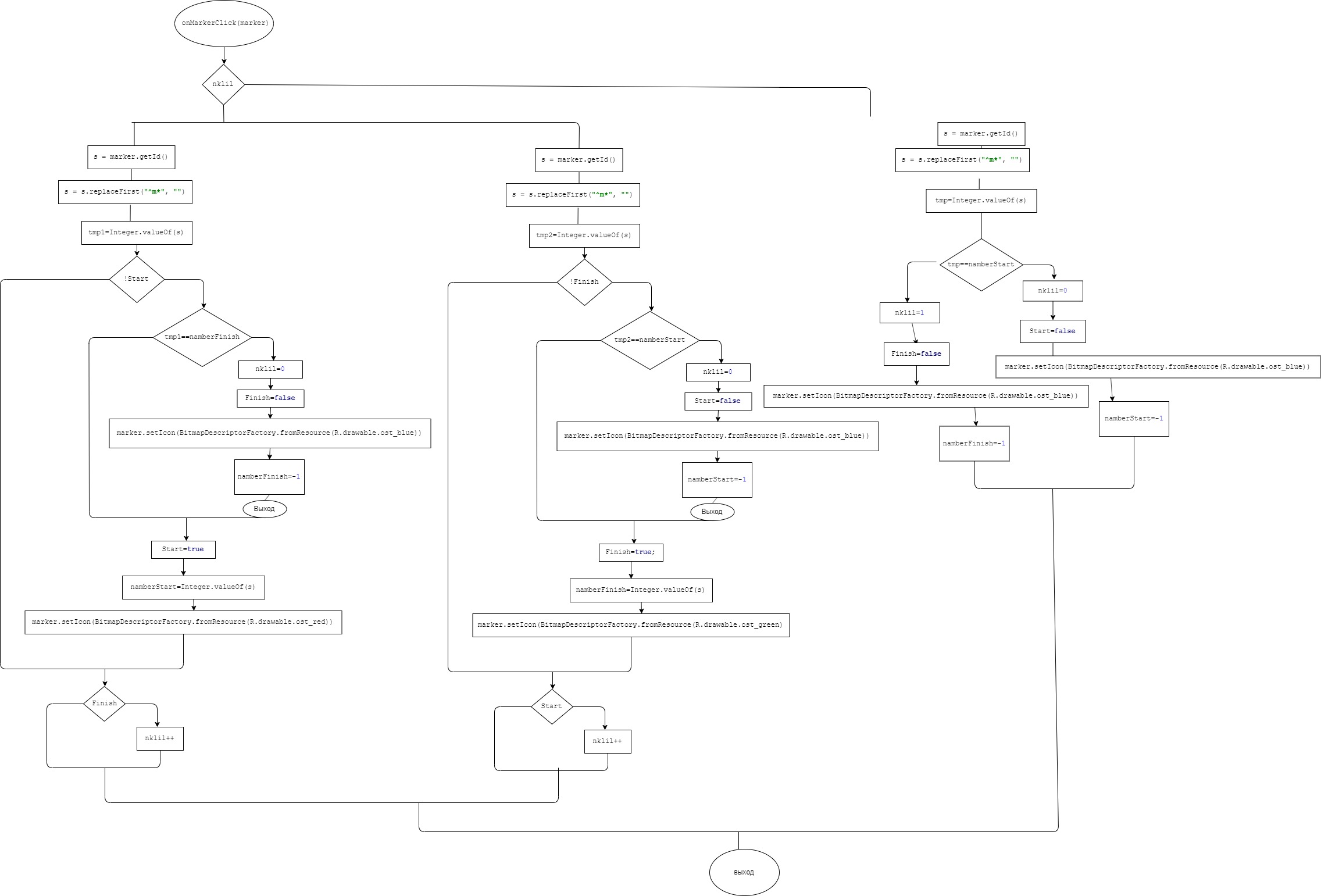
2. Назначение: переключение по пунктам меню



1. Заголовок: public void onMapReady(GoogleMap map)
2. Назначение: выставляет метки остановок на карту и выставляет положение камеры



1. Заголовок: public boolean onMarkerClick(final Marker marker)
2. Назначение: отслеживает точку старта пассажира и конечную точку, отправляет эти данные на сервер



# Тестовые данные

|  |  |
| --- | --- |
| **Исходные данные** | **Ожидаемый результат** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Разбиение программы на модули

Программа состоит из 4 модулей и сохранена.

Модуль MainActivity.java предназначен для авторизации пользователей и входа в приложение.

Модуль registration.java предназначен для регистрации новых пользователей.

Модуль account.java предназначен для настроек аккаунта пользователей.

Модуль Myclientserver.java предназначен для работы с сервером.

Модуль MapActivity.java предназначен для работы с картами.

# Заключение

В ходе выполнения курсовой работы были получены новые знания и умения в сфере разработки алгоритмов, написания и отладки программ для решения прикладных задач с использованием языка Java, а также создания программ, состоящих из нескольких модулей, были получены навыки работы с сервером.

# Список используемой литературы:

1. Шилдт Г. - Java 8. Полное руководство. 9-е издание (2015)
2. Сайт <http://developer.alexanderklimov.ru/android/>
3. Сайт <https://developers.google.com/maps/documentation/android-sdk/marker>
4. Сайт https://javatalks.ru/topics/42851

# Приложения

**package** com.example.streybus;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** java.io.FileInputStream;  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 **private** EditText **username**;  
 **private** EditText **password**;  
 **private** Button **login**;  
 **private** Button **registration**;  
 **private** TextView **loginLocked**;  
 **private** TextView **attempts**;  
 **private** TextView **numberOfAttempts**;  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 **username** = (EditText) findViewById(R.id.***edit\_password***);  
 **password** = (EditText) findViewById(R.id.***edit\_FIO***);  
 **login** = (Button) findViewById(R.id.***button\_login***);  
 **registration** = (Button) findViewById(R.id.***button\_registration***);  
 **numberOfAttempts** = (TextView) findViewById(R.id.***number\_of\_attempts***);  
  
 **try** {  
 log(); *// проверка на вход* } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
**void** log\_write(String t) **throws** IOException {  
 *// запись в файл результата входа* FileOutputStream foz = openFileOutput(**nem**, Context.***MODE\_PRIVATE***);  
 foz.write(t.getBytes());  
 foz.close();  
}  
String **nem**=**"0"**;  
 **void** log () **throws** IOException { *// проверка на вход* **try** {  
 FileInputStream fos = openFileInput(**nem**) ;  
 **byte**[] bytes = **new byte**[fos.available()];  
 fos.read(bytes);  
  
 String text = **new** String (bytes);  
 fos.close();  
 **if** (text.equals(**"1"**)){  
 Intent intent = **new** Intent(MainActivity.**this**, Main2\_menu.**class**);  
 startActivity(intent);  
 }  
 }  
 **catch** (IOException e ) {  
  
 String tt = **"0"**; */\*\* при запуске в первый раз записывпем в файл 0 \*\*/* log\_write(tt);  
 }  
  
 }  
  
  
  
 String **key** = **"0"**; *// ключ входа* String **s**;  
  
 **public void** registration(View view) { *//крнопка регистрации преход на новую активити* Intent intent = **new** Intent(MainActivity.**this**,registraion.**class**);  
 startActivity(intent);  
 }  
  
 **public void** Login(View view) **throws** IOException { *// кнопка входа* **s**=**""**;  
  
 **s**=**username**.getText().toString()+**"#"**+**password**.getText().toString();  
 Myclientserver Myclientserver = **new** Myclientserver(**s**,**key**);  
 Myclientserver.start();  
 **if** (**key**.equals(**"1"**)) {  
 Toast.*makeText*(getApplicationContext(), **"Вход выполнен!"**, Toast.***LENGTH\_SHORT***).show();  
 log\_write(**"1"**);  
 Intent intent = **new** Intent(MainActivity.**this**, Main2\_menu.**class**);  
 startActivity(intent);  
 } **else** {  
 Toast.*makeText*(getApplicationContext(), **"Неправильные данные!"**, Toast.***LENGTH\_SHORT***).show();  
  
  
 }  
 }  
}

**package** com.example.streybus;  
  
**import** java.io.DataInputStream;  
**import** java.io.DataOutputStream;  
**import** java.io.InputStream;  
**import** java.io.OutputStream;  
**import** java.net.InetAddress;  
**import** java.net.Socket;  
  
**class** Myclientserver **extends** Thread {  
  
 **public** Myclientserver(String s, String key) {  
  
 **int** serverPort = 2048; *// здесь обязательно нужно указать порт к которому привязывается сервер.* String address = **"192.168.24.24"**; *// это IP-адрес компьютера, где исполняется наша серверная программа.  
 // Здесь указан адрес того самого компьютера где будет исполняться и клиент.* **try** {  
 InetAddress ipAddress = InetAddress.*getByName*(address); *// создаем объект который отображает вышеописанный IP-адрес.  
 // System.out.println("Any of you heard of a socket with IP address " + address + " and port " + serverPort + "?");* Socket socket = **new** Socket(ipAddress, serverPort); *// создаем сокет используя IP-адрес и порт сервера.  
 // System.out.println("Yes! I just got hold of the program.");  
  
 // Берем входной и выходной потоки сокета, теперь можем получать и отсылать данные клиентом.* InputStream sin = socket.getInputStream();  
 OutputStream sout = socket.getOutputStream();  
  
 *// Конвертируем потоки в другой тип, чтоб легче обрабатывать текстовые сообщения.* DataInputStream in = **new** DataInputStream(sin);  
 DataOutputStream out = **new** DataOutputStream(sout);  
  
 *// Создаем поток для чтения с клавиатуры.* out.writeBytes(s); *// отсылаем строку текста серверу.* out.flush(); *// заставляем поток закончить передачу данных.* key = in.readUTF(); *// ответ сервера* System.***out***.println();  
  
 }  
 **catch** (Exception x) {  
 x.printStackTrace();  
 System.***out***.print(**"Ошибка"**);  
 }  
  
  
 }  
  
}

**package** com.example.streybus;  
  
  
**import** android.os.Bundle;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
  
**import** androidx.appcompat.app.ActionBarDrawerToggle;  
**import** androidx.appcompat.app.AppCompatActivity;  
**import** androidx.appcompat.widget.Toolbar;  
**import** androidx.core.view.GravityCompat;  
**import** androidx.drawerlayout.widget.DrawerLayout;  
**import** androidx.fragment.app.Fragment;  
**import** androidx.fragment.app.FragmentManager;  
**import** androidx.navigation.NavController;  
**import** androidx.navigation.Navigation;  
**import** androidx.navigation.ui.AppBarConfiguration;  
**import** androidx.navigation.ui.NavigationUI;  
  
**import** com.example.streybus.fragments.account.accountFragment;  
**import** com.example.streybus.fragments.information.informationFragment;  
**import** com.example.streybus.fragments.map.mapFragment;  
**import** com.example.streybus.fragments.settings.settingsFragment;  
**import** com.example.streybus.fragments.support.supportFragment;  
**import** com.google.android.material.navigation.NavigationView;  
  
  
**public class** Main2\_menu **extends** AppCompatActivity **implements** NavigationView.OnNavigationItemSelectedListener{  
  
 **private** AppBarConfiguration **mAppBarConfiguration**;  
 accountFragment **faccount**;  
 mapFragment **fmap**;  
 settingsFragment **fsettings**;  
 supportFragment **fsupport**;  
 informationFragment **finformation**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main2\_menu***);  
 Toolbar toolbar = findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 DrawerLayout drawer =(DrawerLayout) findViewById(R.id.***drawer\_layout***);  
 ActionBarDrawerToggle toggle = **new** ActionBarDrawerToggle(  
 **this**, drawer, toolbar, R.string.***navigation\_drawer\_open***, R.string.***navigation\_drawer\_close***);  
 drawer.setDrawerListener(toggle);  
 toggle.syncState();  
 NavigationView navigationView =(NavigationView) findViewById(R.id.***nav\_view***);  
 *// Passing each menu ID as a set of Ids because each  
 // menu should be considered as top level destinations.* **mAppBarConfiguration** = **new** AppBarConfiguration.Builder(  
 R.id.***nav\_map***, R.id.***nav\_support***, R.id.***nav\_settings***,  
 R.id.***nav\_information***)  
 .setDrawerLayout(drawer)  
 .build();  
 NavController navController = Navigation.*findNavController*(**this**, R.id.***nav\_host\_fragment***);  
 NavigationUI.*setupActionBarWithNavController*(**this**, navController, **mAppBarConfiguration**);  
 NavigationUI.*setupWithNavController*(navigationView, navController);  
  
  
  
 navigationView.setNavigationItemSelectedListener(**this**);  
 **faccount** = **new** accountFragment();  
 **fmap** = **new** mapFragment();  
 **fsettings** = **new** settingsFragment();  
 **fsupport** = **new** supportFragment();  
 **finformation** = **new** informationFragment();  
  
 }  
 @Override  
 **public void** onBackPressed() {  
 DrawerLayout drawer = (DrawerLayout) findViewById(R.id.***drawer\_layout***);  
 **if** (drawer.isDrawerOpen(GravityCompat.***START***)) {  
 drawer.closeDrawer(GravityCompat.***START***);  
 } **else** {  
 **super**.onBackPressed();  
 }  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 *// Inflate the menu; this adds items to the action bar if it is present.* **return true**;  
 }  
  
 @Override  
 **public boolean** onSupportNavigateUp() {  
 NavController navController = Navigation.*findNavController*(**this**, R.id.***nav\_host\_fragment***);  
 **return** NavigationUI.*navigateUp*(navController, **mAppBarConfiguration**)  
 || **super**.onSupportNavigateUp();  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 *// Handle action bar item clicks here. The action bar will  
 // automatically handle clicks on the Home/Up button, so long  
 // as you specify a parent activity in AndroidManifest.xml.* **int** id = item.getItemId();  
  
 *//noinspection SimplifiableIfStatement* **return super**.onOptionsItemSelected(item);  
 }  
  
  
  
 @SuppressWarnings(**"StatementWithEmptyBody"**)  
 @Override  
 **public boolean** onNavigationItemSelected(MenuItem item) {  
  
 *// Создадим новый фрагмент* Fragment fragment = **null**;  
 Class fragmentClass = **null**;  
  
 *// Handle navigation view item clicks here.* **int** id = item.getItemId();  
  
 **if** (id == R.id.***nav\_map***) {  
  
 fragmentClass = mapFragment.**class**;  
  
 } **else if** (id == R.id.***nav\_account***) {  
  
 fragmentClass = accountFragment.**class**;  
 } **else if** (id == R.id.***nav\_support***) {  
 fragmentClass = supportFragment.**class**;  
 } **else if** (id == R.id.***nav\_settings***) {  
 fragmentClass = settingsFragment.**class**;  
 } **else if** (id == R.id.***nav\_information***) {  
 fragmentClass = informationFragment.**class**;  
 }  
  
 **try** {  
 fragment = (Fragment) fragmentClass.newInstance();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
  
 *// Вставляем фрагмент, заменяя текущий фрагмент* FragmentManager fragmentManager = getSupportFragmentManager();  
 fragmentManager.beginTransaction().replace(R.id.***container***, fragment).commit();  
 *// Выделяем выбранный пункт меню в шторке* item.setChecked(**true**);  
 *// Выводим выбранный пункт в заголовке* setTitle(item.getTitle());  
  
 DrawerLayout drawer = (DrawerLayout) findViewById(R.id.***drawer\_layout***);  
 drawer.closeDrawer(GravityCompat.***START***);  
 **return true**;  
 }  
}

**package** com.example.streybus;  
  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
  
**public class** registraion **extends** AppCompatActivity {  
  
 **private** EditText **login**;  
 **private** EditText **password**;  
 **private** EditText **FIO**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_registraion***);  
 **login** = (EditText) findViewById(R.id.***edit\_login***);  
 **password** = (EditText) findViewById(R.id.***edit\_password***);  
 **FIO** = (EditText) findViewById(R.id.***edit\_FIO***);  
 }  
  
 String **nem**=**"0"**;  
 **void** log\_write(String t) **throws** IOException {  
 *// запись в файл результата входа* FileOutputStream foz = openFileOutput(**nem**, Context.***MODE\_PRIVATE***);  
 foz.write(t.getBytes());  
 foz.close();  
 }  
 **public void** registred(View view){  
 String l= (**login**.getText().toString());  
 String P=(**password**.getText().toString());  
 String F = (**FIO**.getText().toString());  
 String key = **null**;  
 String s=**""**;  
 s=l+**"#"**+P+**"#"**+F;  
 Myclientserver Myclientserver = **new** Myclientserver(s,key);  
 Myclientserver.start();  
 **if** (key.equals(**"1"**)){  
  
 Toast.*makeText*(getApplicationContext(), **"Регистрация прошла успешно!"**,Toast.***LENGTH\_SHORT***).show();  
 **try** {  
 log\_write(**"1"**);  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 Intent intent = **new** Intent(registraion.**this**,MainActivity.**class**);  
 startActivity(intent);}  
 **else** {  
 Toast.*makeText*(getApplicationContext(), **"Ошибка регистрации "**,Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
}

**import** android.Manifest;  
**import** android.annotation.SuppressLint;  
**import** android.content.pm.PackageManager;  
**import** android.graphics.Color;  
**import** android.os.AsyncTask;  
**import** android.os.Build;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v4.app.FragmentActivity;  
**import** android.os.Bundle;  
  
**import** java.io.BufferedReader;  
**import** java.io.ByteArrayOutputStream;  
**import** java.io.DataInputStream;  
**import** java.io.DataOutputStream;  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.io.InputStreamReader;  
**import** java.io.OutputStream;  
**import** java.io.PrintWriter;  
**import** java.net.HttpURLConnection;  
**import** java.net.InetAddress;  
**import** java.net.MalformedURLException;  
**import** java.net.Socket;  
**import** java.net.SocketTimeoutException;  
**import** java.net.URL;  
  
**import** com.google.android.gms.common.api.ApiException;  
**import** com.google.android.gms.maps.CameraUpdate;  
**import** com.google.android.gms.maps.CameraUpdateFactory;  
**import** com.google.android.gms.maps.GoogleMap;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.SupportMapFragment;  
**import** com.google.android.gms.maps.model.BitmapDescriptorFactory;  
**import** com.google.android.gms.maps.model.CameraPosition;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.google.android.gms.maps.model.LatLngBounds;  
**import** com.google.android.gms.maps.model.Marker;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
**import** com.google.android.gms.maps.model.Polyline;  
**import** com.google.android.gms.maps.model.PolylineOptions;  
**import** com.google.maps.DirectionsApi;  
**import** com.google.maps.GeoApiContext;  
**import** com.google.maps.model.DirectionsResult;  
  
**import** android.app.Service;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.os.Bundle;  
**import** android.os.IBinder;  
**import** android.util.Log;  
**import** android.view.Gravity;  
**import** android.widget.Toast;  
  
**import** java.net.UnknownHostException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**public class** MapsActivity **extends** FragmentActivity **implements** GoogleMap.OnMarkerClickListener,  
 OnMapReadyCallback {  
  
  
 **private** List<LatLng> ST = **new** ArrayList<>();  
 Marker[] mSt;  
 GoogleMap mMap;  
  
  
 @SuppressLint(**"MissingPermission"**)  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_maps);  
  
  
  
*// координаты остановок* ST.add(**new** LatLng(50.578377, 36.591314));  
 ST.add(**new** LatLng(50.578249, 36.592727));  
 ST.add(**new** LatLng(50.578481, 36.586030));  
 ST.add(**new** LatLng(50.578143, 36.587047));  
 ST.add(**new** LatLng(50.575265, 36.582893));  
 ST.add(**new** LatLng(50.575849, 36.583160));  
 ST.add(**new** LatLng(50.579013, 36.580314));  
 ST.add(**new** LatLng(50.578733, 36.581149));  
 mSt = **new** Marker[ST.size()];  
  
  
 *// Получите SupportMapFragment и получите уведомление, когда карта будет готова к использованию.* SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.map);  
 mapFragment.getMapAsync(**this**);  
  
 }  
  
  
  
  
  
  
  
 @Override  
 **public void** onMapReady(GoogleMap map) {  
 mMap = map;  
*//cавим метки* **for** (**int** i = 0; i < ST.size(); i++) {  
 mSt[i] =map.addMarker(**new** MarkerOptions()  
 .position(ST.get(i))  
 .icon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_blue)));  
 }  
 mMap.setOnMarkerClickListener(**this**);  
  
*/\*\*  
 //рисуем граф  
  
 PolylineOptions options = new PolylineOptions().width(10).color(Color.BLUE).geodesic(true);  
 for (int z=0;z<ST.size();z++) {  
 LatLng point = ST.get(z);  
 for (int i = 0; i < ST.size(); i++) {  
 options.add(point);  
 point = ST.get(i);  
 options.add(point);  
 point = ST.get(z);  
  
 }  
 }  
 mMap.addPolyline(options);  
  
  
  
\*\*/  
  
  
//ставим камеру* CameraPosition cameraPosition = **new** CameraPosition.Builder()  
 .target(**new** LatLng(50.599761, 36.601329)) *//координаты города* .zoom(14) *// приближение* .bearing(45) *// угол поворота от севера по часовой* .tilt(5) *// уровень наклона камеры* .build();  
 CameraUpdate cameraUpdate = CameraUpdateFactory.newCameraPosition(cameraPosition);  
 map.animateCamera(cameraUpdate);  
 }  
  
  
 String s;  
 **int** nklil=0;  
 **int** namberStart=-1;  
 **boolean** Start=**false**;  
 **boolean** Finish=**false**;  
 **int** namberFinish=-1;  
 **int** tmp;  
 @Override  
 **public boolean** onMarkerClick(**final** Marker marker) {  
 nklil++;  
**switch** (nklil){  
 **case** 1:  
 s = marker.getId();  
 s = s.replaceFirst(**"^m\*"**, **""**);  
 **int** tmp1=Integer.valueOf(s);  
 **if** (!Start) {  
 **if** (tmp1==namberFinish){  
 nklil=0;  
 Finish=**false**;  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_blue));  
 namberFinish=-1;  
 **break**;  
 }  
 Start=**true**;  
 namberStart=Integer.valueOf(s);  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_red));  
 }  
 **if** (Finish){  
 nklil++;  
 }  
  
 **break**;  
 **case** 2:  
 s = marker.getId();  
 s = s.replaceFirst(**"^m\*"**, **""**);  
 **int** tmp2=Integer.valueOf(s);  
 **if** (!Finish) {  
 **if**(tmp2==namberStart){  
 nklil=0;  
 Start=**false**;  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_blue));  
 namberStart=-1;  
 **break**;  
 }  
 namberFinish=Integer.valueOf(s);  
 Finish=**true**;  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_green));  
 }  
 **if** (Start){  
 nklil++;  
 }  
 **break**;  
  
 **default**:  
 s = marker.getId();  
 s = s.replaceFirst(**"^m\*"**, **""**);  
 tmp=Integer.valueOf(s);  
 **if**(tmp==namberStart) {  
 nklil=0;  
 Start=**false**;  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_blue));  
 namberStart=-1;  
 }  
 **else if** (tmp==namberFinish){  
 nklil=1;  
 Finish=**false**;  
 marker.setIcon(BitmapDescriptorFactory.fromResource(R.drawable.ost\_blue));  
 namberFinish=-1;  
 }  
 **break**;  
  
}  
  
  
Myclientserver Myclientserver = **new** Myclientserver(s,key);  
Myclientserver.start();

**return true**;  
  
}}