Classe Principal

```
<?xml version="1.0" encoding="utf-8"?>

 // logar usando conta google service;

<com.google.android.gms.common.SignInButton
 android:id="@+id/sign_in_button"
android:layout_width="wrap_content"
 android:layout_height="wrap_content" />
@Override protected void onCreate(Bundle savedInstanceState)
{
   super.onCreate(savedInstanceState);
mPlusClient = new PlusClient.Builder(this, this,this)
.setVisibleActivities("http://schemas.google.com/AddActivity", "http://schemas.google.com/BuyActivity") .build();
}
findViewById(R.id.sign_in_button).setOnClickListener(this);
@Override
 public void onClick(View view){
 if (view.getId() == R.id.sign_in_button && !mPlusClient.isConnected()) {
if (mConnectionResult == null) {
mConnectionProgressDialog.show(); }
else {
 try { mConnectionResult.startResolutionForResult(this, REQUEST_CODE_RESOLVE_ERR); }
catch (SendIntentException e) {
 // Tente se conectar novamente.
mConnectionResult = null; mPlusClient.connect(); }
}
     }
}
@Override
 public void onConnected(Bundle connectionHint) {
mConnectionProgressDialog.dismiss();
Toast.makeText(this, "User is connected!", Toast.LENGTH_LONG).show();
 }
@Override
 public void onClick(View view) {
 if (view.getId() == R.id.sign_out_button) {
if (mPlusClient.isConnected()) {
mPlusClient.clearDefaultAccount();
 mPlusClient.disconnect();
mPlusClient.connect(); }
 return true;
}
 }
<Button
android:id="@+id/button"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
 android:layout_centerHorizontal="true"
 android:text="@string/common_signin_button_text_long" />
```

```
// configuração da tela principal do aplicativo e lista com permissões;
<manifest android:versionCode="5" android:versionName="1.1.01" android:installLocation="auto"
package="com.dej.bustime"
  xmlns:android="http://schemas.android.com/apk/res/android">
// permissões de uso de dados do celular;
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
// permissões de uso do gps do celular;
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_MOCK_LOCATION"/>
    <supports-screens android:anyDensity="true" android:xlargeScreens="false" />
    <application android:theme="@style/Theme.Mc" android:label="@string/app_name"
android:icon="@drawable/ic_launcher" android:name=".BusTimeApplication"
android:configChanges="keyboardHidden|orientation" android:allowBackup="false">
        <activity android:theme="@style/Theme.Sherlock.Light.NoActionBar" android:label="@string/main_label"
android:name="com.dej.bustime.SplashActivity" android:screenOrientation="portrait"
android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:theme="@style/Theme.Sherlock.Light.NoActionBar" android:label="@string/app_name"
android:name="com.dej.gurpo7.MainActivity" android:launchMode="singleTask"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation"
android:windowSoftInputMode="adjustPan">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:label="@string/favoritos" android:name="com.dej.bustime.favoritos.FavoritosActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:label="@string/tela_bus" android:name="com.dej.grupo7.detalhes.BusTimeActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
            </intent-filter>
        </activity>
        <activity android:label="@string/sobre_bustime" android:name="com.dej.grupo7.SobreActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation"
android:uiOptions="0x1">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
```

```
            </intent-filter>
        </activity>
        <activity android:label="@string/termos" android:name="com.dej.gurpo7.TermosActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:theme="@style/Theme.Sherlock.Light.NoActionBar" android:label="@string/bemvindo"
android:name="com.dej.grupo7.saudacoes.SaudacoesActivity" android:screenOrientation="portrait"
android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:label="@string/ajuda" android:name="com.dej.grupo7.ajuda.AjudaActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:label="@string/resultados" android:name="com.dej.grupo7.favoritos.ResultadosActivity"
android:screenOrientation="portrait" android:configChanges="keyboardHidden|orientation">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <meta-data android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version" />
        <activity android:name="com.google.android.gms.ads.AdActivity"
android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|uiMode" />

    </application>
</manifest>
```

Classe Ajuda

```
package com.dej.grupo7.ajuda;

import android.content.Intent;
import android.content.res.Resources;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentTransaction;
import android.support.v4.view.ViewPager;
import android.support.v4.view.ViewPager.SimpleOnPageChangeListener;
```

```
import com.actionbarsherlock.app.ActionBar;
import com.actionbarsherlock.app.ActionBar.Tab;
import com.actionbarsherlock.app.ActionBar.TabListener;
import com.actionbarsherlock.app.SherlockFragmentActivity;
import com.actionbarsherlock.view.MenuItem;
import com.dej.bustime.MainActivity;
import com.dej.bustime.views.mViewPager;
import com.google.analytics.tracking.android.EasyTracker;
import com.google.android.gms.ads.AdRequest.Builder;
import com.google.android.gms.ads.AdView;

public class AjudaActivity
  extends SherlockFragmentActivity
  implements ActionBar.TabListener
{
  ActionBar mActionBar;
  ViewPager mPager;

  protected void onCreate(Bundle paramBundle)
  {
    super.onCreate(paramBundle);
    setContentView(2130903061);
    this.mActionBar = getSupportActionBar();
    this.mActionBar.setDisplayHomeAsUpEnabled(true);
    this.mActionBar.setDisplayShowTitleEnabled(true);
    this.mActionBar.setNavigationMode(2);
    this.mActionBar.setIcon(getResources().getDrawable(2130837658));
    this.mActionBar.setBackgroundDrawable(getResources().getDrawable(2130837623));
    this.mPager = ((mViewPager)findViewById(2131099712));
    FragmentManager localFragmentManager = getSupportFragmentManager();
    ViewPager.SimpleOnPageChangeListener local1 = new ViewPager.SimpleOnPageChangeListener()
    {
      public void onPageSelected(int paramAnonymousInt)
      {
        super.onPageSelected(paramAnonymousInt);
        AjudaActivity.this.mActionBar.setSelectedNavigationItem(paramAnonymousInt);
      }
    };
    this.mPager.setOnPageChangeListener(local1);
    PagerAdapterAjuda localPagerAdapterAjuda = new PagerAdapterAjuda(localFragmentManager);
    this.mPager.setAdapter(localPagerAdapterAjuda);
    this.mPager.setOffscreenPageLimit(2);
    ActionBar.Tab localTab1 = this.mActionBar.newTab().setText("PESQUISAS").setTabListener(this);
    this.mActionBar.addTab(localTab1);
    ActionBar.Tab localTab2 = this.mActionBar.newTab().setText("LINHAS").setTabListener(this);
    this.mActionBar.addTab(localTab2);
    ActionBar.Tab localTab3 = this.mActionBar.newTab().setText("FAVORITOS").setTabListener(this);
    this.mActionBar.addTab(localTab3);
    ConnectivityManager localConnectivityManager = (ConnectivityManager)getSystemService("connectivity");
    if ((localConnectivityManager.getActiveNetworkInfo() != null) &&
(localConnectivityManager.getActiveNetworkInfo().isConnectedOrConnecting()))
```

```
    {
      AdView localAdView = (AdView)findViewById(2131099713);
      localAdView.loadAd(new
AdRequest.Builder().addTestDevice("00B515006C90F80334C1E3CA4E6A4CC9").build());
      localAdView.setVisibility(0);
    }
  }

  public boolean onOptionsItemSelected(MenuItem paramMenuItem)
  {
    switch (paramMenuItem.getItemId())
    {
    default:
      return super.onOptionsItemSelected(paramMenuItem);
    }
    Intent localIntent = new Intent(this, MainActivity.class);
    localIntent.addFlags(67108864);
    startActivity(localIntent);
    finish();
    return true;
  }

  public void onStart()
  {
    super.onStart();
    EasyTracker.getInstance(this).activityStart(this);
  }

  public void onStop()
  {
    super.onStop();
    EasyTracker.getInstance(this).activityStop(this);
  }

  public void onTabReselected(ActionBar.Tab paramTab, FragmentTransaction paramFragmentTransaction) {}

  public void onTabSelected(ActionBar.Tab paramTab, FragmentTransaction paramFragmentTransaction)
  {
    this.mPager.setCurrentItem(paramTab.getPosition());
  }

  public void onTabUnselected(ActionBar.Tab paramTab, FragmentTransaction paramFragmentTransaction) {}
}

package com.dej.bustime.db;

import android.content.Context;
import android.content.res.AssetManager;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
```

```
import android.database.sqlite.SQLiteException;
import android.database.sqlite.SQLiteOpenHelper;
import com.dej.bustime.BusTimeApplication;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
```

Classe Favoritos

```
public class DatabaseFavoritosHelper
  extends SQLiteOpenHelper
{
  private static String DB_NAME = "BusTime.fav.db";
  private static String DB_PATH = BusTimeApplication.getAppContext().getFilesDir().getParentFile().getPath() +
"/databases/";
  private final Context myContext;
  private SQLiteDatabase myDataBase;

  public DatabaseFavoritosHelper(Context paramContext)
  {
    super(paramContext, DB_NAME, null, 1);
    this.myContext = paramContext;
  }

  private boolean checkDataBase()
  {
    try
    {
      SQLiteDatabase localSQLiteDatabase2 = SQLiteDatabase.openDatabase(DB_PATH + DB_NAME, null, 1);
      localSQLiteDatabase1 = localSQLiteDatabase2;
    }
    catch (SQLiteException localSQLiteException)
    {
      for (;;)
      {
        SQLiteDatabase localSQLiteDatabase1 = null;
      }
    }
    if (localSQLiteDatabase1 != null) {
      localSQLiteDatabase1.close();
    }
    return localSQLiteDatabase1 != null;
  }

  private void copyDataBase()
    throws IOException
```

```
  {
    InputStream localInputStream = getMyContext().getAssets().open(DB_NAME);
    FileOutputStream localFileOutputStream = new FileOutputStream(DB_PATH + DB_NAME);
    byte[] arrayOfByte = new byte[1024];
    for (;;)
    {
      int i = localInputStream.read(arrayOfByte);
      if (i <= 0)
      {
        localFileOutputStream.flush();
        localFileOutputStream.close();
        localInputStream.close();
        return;
      }
      localFileOutputStream.write(arrayOfByte, 0, i);
    }
  }

  public void close()
  {
    try
    {
      if (this.myDataBase != null) {
        this.myDataBase.close();
      }
      super.close();
      return;
    }
    finally {}
  }

  public void createDataBase()
    throws IOException
  {
    if (checkDataBase())
    {
      ArrayList localArrayList = new ArrayList();
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha ORDER BY NR_linha ASC",
null);
      if (localCursor.moveToFirst()) {
        do
        {
          localArrayList.add(localCursor.getString(localCursor.getColumnIndex("NR_linha")));
        } while (localCursor.moveToNext());
      }
      localSQLiteDatabase.close();
      TableLinhasFavController localTableLinhasFavController = new TableLinhasFavController(this.myContext);
      Iterator localIterator = localArrayList.iterator();
      for (;;)
      {
```

```
        if (!localIterator.hasNext()) {
          return;
        }
        String str = (String)localIterator.next();
        if (str.equals("2601"))
        {
          localTableLinhasFavController.delete(str);
          localTableLinhasFavController.add(str + "/1", "A 100 - Rodoviária / Terminal Central");
        }
        else if (str.equals("6201"))
        {
          localTableLinhasFavController.delete(str);
          localTableLinhasFavController.add(str + "/1", "T 151 - Terminal Industrial / Terminal Central");
        }
      }
    }
    getReadableDatabase();
    try
    {
      copyDataBase();
      return;
    }
    catch (IOException localIOException)
    {
      throw new Error("Error copying database fav");
    }
  }

  public Context getMyContext()
  {
    return this.myContext;
  }

  public void onCreate(SQLiteDatabase paramSQLiteDatabase) {}

  public void onUpgrade(SQLiteDatabase paramSQLiteDatabase, int paramInt1, int paramInt2) {}

  public void openDataBase()
    throws SQLException
  {
    this.myDataBase = SQLiteDatabase.openDatabase(DB_PATH + DB_NAME, null, 1);
  }
}

package com.dej.bustime.db;

import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import com.dej.bustime.BusTime;

import java.util.ArrayList;
```

```
import java.util.Collections;
import java.util.Comparator;
import java.util.List;
```

Controle de linhas: “tabela de favoritos”

```
public class TableLinhasController
  extends DatabaseLinhasHelper
{
  public static final String HORARIO_IDA_DIA_UTIL = "HR_ida_dia_util";
  public static final String HORARIO_IDA_DOMINGO = "HR_ida_dom";
  public static final String HORARIO_IDA_SABADO = "HR_ida_sab";
  public static final String HORARIO_VOLTA_DIA_UTIL = "HR_volta_dia_util";
  public static final String HORARIO_VOLTA_DOMINGO = "HR_volta_dom";
  public static final String HORARIO_VOLTA_SABADO = "HR_volta_sab";
  public static final String ID = "_id";
  public static final String IDA = "DS_ida";
  public static final String INT_IDA_IDA = "DS_int_ida_ida";
  public static final String INT_IDA_VOLTA = "DS_int_ida_volta";
  public static final String INT_VOLTA_IDA = "DS_int_volta_ida";
  public static final String INT_VOLTA_VOLTA = "DS_int_volta_volta";
  public static final String ITINERARIO_IDA = "DS_iti_ida";
  public static final String ITINERARIO_VOLTA = "DS_iti_volta";
  public static final String NOME = "NM_linha";
  public static final String NUM = "NR_linha";
  public static final String PONTOS = "DS_pontos";
  public static final String TABLE_NAME = "tb_linha";
  public static final String VOLTA = "DS_volta";

  public TableLinhasController(Context paramContext)
  {
    super(paramContext);
  }

  public int countRecords()
  {
    SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
    int i = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha", null).getCount();
    localSQLiteDatabase.close();
    return i;
  }

  /* Error */
  public BusTime findOne(String paramString)
  {

public List<BusTime> getAll()
  {
    ArrayList localArrayList = new ArrayList();
    try
```

```
    {
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha", null);
      if (localCursor.moveToFirst()) {
        do
        {
          localArrayList.add(new BusTime(localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha"))));
        } while (localCursor.moveToNext());
      }
      localCursor.close();
      localSQLiteDatabase.close();
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
    return localArrayList;
  }

  public List<BusTime> search(String paramString)
  {
    ArrayList localArrayList = new ArrayList();
    try
    {
      String str = "SELECT * FROM tb_linha WHERE NR_linha LIKE '" + paramString + "%' OR " + "NM_linha" +
" LIKE '%" + paramString + "%' ORDER BY " + "NR_linha" + " ASC";
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery(str, null);
      if (localCursor.moveToFirst()) {
        do
        {
          localArrayList.add(new BusTime(localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha"))));
        } while (localCursor.moveToNext());
      }
      localCursor.close();
      localSQLiteDatabase.close();
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
    return localArrayList;
  }

  public List<BusTime> searchByOneStreet(String paramString)
```

```
  {
    ArrayList localArrayList = new ArrayList();
    try
    {
      String str = "SELECT * FROM tb_linha WHERE DS_iti_ida LIKE '%" + paramString + "%' OR " +
"DS_iti_volta" + " LIKE '%" + paramString + "%' ORDER BY " + "NR_linha" + " ASC";
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery(str, null);
      if (localCursor.moveToFirst()) {
        do
        {
          localArrayList.add(new Bustime(localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha"))));
        } while (localCursor.moveToNext());
      }
      localCursor.close();
      localSQLiteDatabase.close();
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
    return localArrayList;
  }

  public List<BusTime> searchByTwoStreet(String paramString1, String paramString2)
  {
    ArrayList localArrayList = new ArrayList();
    try
    {
      String str = "SELECT * FROM tb_linha WHERE (DS_iti_ida LIKE '%" + paramString1 + "%' OR " +
"DS_iti_volta" + " LIKE '%" + paramString1 + "%') AND (" + "DS_iti_ida" + " LIKE '%" + paramString2 + "%' OR
" + "DS_iti_volta" + " LIKE '%" + paramString2 + "%') ORDER BY " + "NR_linha" + " ASC";
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery(str, null);
      if (localCursor.moveToFirst()) {
        do
        {
          localArrayList.add(new BusTime(localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha"))));
        } while (localCursor.moveToNext());
      }
      localCursor.close();
      localSQLiteDatabase.close();
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
```

```
      localNullPointerException.printStackTrace();
    }
    return localArrayList;
  }

  public List<BusTime> simpleFindMany(String[] paramArrayOfString)
  {
    localArrayList = new ArrayList();
    try
    {
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      int i = paramArrayOfString.length;
      for (int j = 0;; j++)
      {
        if (j >= i)
        {
          Collections.sort(localArrayList, new Comparator()
          {
            public int compare(BusTime paramAnonymousBusTime1, BusTime paramAnonymousBusTime2)
            {
              return Integer.parseInt(paramAnonymousBusTime1.getNumLinha().substring(0, 4)) -
Integer.parseInt(paramAnonymousBusTime2.getNumLinha().substring(0, 4));
            }
          });
          localSQLiteDatabase.close();
          return localArrayList;
        }
        String str = paramArrayOfString[j];
        Cursor localCursor = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha WHERE NR_linha = '" + str
+ "' ", null);
        if (localCursor.moveToFirst()) {
          localArrayList.add(new BusTime (localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha"))));
        }
        localCursor.close();
      }
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
  }

  public BusTime simpleFindOne(String paramString)
  {
    String str = "SELECT * FROM tb_linha WHERE NR_linha = '" + paramString + "' ";
    SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
    Cursor localCursor = localSQLiteDatabase.rawQuery(str, null);
    boolean bool = localCursor.moveToFirst();
```

```
    BusTime localBusTime = null;
    if (bool) {
      localBusTime = new BusTime (localCursor.getInt(localCursor.getColumnIndex("_id")),
localCursor.getString(localCursor.getColumnIndex("NR_linha")),
localCursor.getString(localCursor.getColumnIndex("NM_linha")));
    }
    localCursor.close();
    localSQLiteDatabase.close();
    return localBusTime;
  }
}

package com.dej.bustime.db;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import com.dej.bustime.BusTime;
import java.util.ArrayList;
import java.util.List;

public class TableLinhasFavController
  extends DatabaseFavoritosHelper
{
  public static final String ID = "_id";
  public static final String NOME = "NM_linha";
  public static final String NUM = "NR_linha";
  public static final String TABLE_NAME = "tb_linha";

  public TableLinhasFavController(Context paramContext)
  {
    super(paramContext);
  }

  public boolean add(String paramString1, String paramString2)
  {
    try
    {
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      ContentValues localContentValues = new ContentValues();
      localContentValues.put("NR_linha", paramString1);
      localContentValues.put("NM_linha", paramString2);
      localSQLiteDatabase.insert("tb_linha", null, localContentValues);
      localSQLiteDatabase.close();
      return true;
    }
    catch (NullPointerException localNullPointerException)
    {
      for (;;)
      {
```

```
      localNullPointerException.printStackTrace();
    }
  }
}

public int countRecords()
{
  SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
  int i = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha", null).getCount();
  localSQLiteDatabase.close();
  return i;
}

public boolean delete(String paramString)
{
  try
  {
    SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
    localSQLiteDatabase.delete("tb_linha", "NR_linha ='" + paramString + "'", null);
    localSQLiteDatabase.close();
    return true;
  }
  catch (NullPointerException localNullPointerException)
  {
    localNullPointerException.printStackTrace();
  }
  return false;
}

public void deleteAll()
{
  try
  {
    SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
    localSQLiteDatabase.execSQL("DELETE FROM tb_linha");
    localSQLiteDatabase.close();
    return;
  }
  catch (NullPointerException localNullPointerException)
  {
    localNullPointerException.printStackTrace();
  }
}

public boolean exists(String paramString)
{
  bool = false;
  try
  {
    String str = "SELECT * FROM tb_linha WHERE NR_linha = '" + paramString + "' ";
    SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
```

```
      if (localSQLiteDatabase.rawQuery(str, null).getCount() > 0) {}
      for (bool = true;; bool = false)
      {
        localSQLiteDatabase.close();
        return bool;
      }
      return bool;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
  }

  public List<BusTime> getAll()
  {
    ArrayList localArrayList = new ArrayList();
    try
    {
      SQLiteDatabase localSQLiteDatabase = getWritableDatabase();
      Cursor localCursor = localSQLiteDatabase.rawQuery("SELECT * FROM tb_linha ORDER BY NR_linha ASC",
null);
      if (localCursor.moveToFirst())
      {
        TableLinhasController localTableLinhasController = new TableLinhasController(getMyContext());
        do
        {

localArrayList.add(localTableLinhasController.simpleFindOne(localCursor.getString(localCursor.getColumnIndex("
NR_linha"))));
        } while (localCursor.moveToNext());
      }
      localSQLiteDatabase.close();
      return localArrayList;
    }
    catch (NullPointerException localNullPointerException)
    {
      localNullPointerException.printStackTrace();
    }
    return localArrayList;
  }
}

package com.dej.bustime;

public class BusTime
{
  private String descIda;
  private String descVolta;
  private String horarioIdaDiaUtil;
  private String horarioIdaDom;
```

```
 private String horarioIdaSab;
 private String horarioVoltaDiaUtil;
 private String horarioVoltaDom;
 private String horarioVoltaSab;
 private int idLinha;
 private String integracaoIdaIda;
 private String integracaoIdaVolta;
 private String integracaoVoltaIda;
 private String integracaoVoltaVolta;
 private String itinerarioIda;
 private String itinerarioVolta;
 private String nomeLinha;
 private String numLinha;
 private String pontos;

 public BusTime(int paramInt, String paramString1, String paramString2)
 {
  this.idLinha = paramInt;
  this.numLinha = paramString1;
  this.nomeLinha = paramString2;
 }

 public BusTime(int paramInt, String paramString1, String paramString2, String paramString3, String paramString4,
String paramString5, String paramString6, String paramString7, String paramString8, String paramString9, String
paramString10, String paramString11, String paramString12, String paramString13, String paramString14, String
paramString15, String paramString16, String paramString17)
 {
  this.idLinha = paramInt;
  this.numLinha = paramString1;
  this.nomeLinha = paramString2;
  this.itinerarioIda = paramString3;
  this.itinerarioVolta = paramString4;
  this.descIda = paramString5;
  this.descVolta = paramString6;
  this.horarioIdaDiaUtil = paramString7;
  this.horarioIdaSab = paramString8;
  this.horarioIdaDom = paramString9;
  this.horarioVoltaDiaUtil = paramString10;
  this.horarioVoltaSab = paramString11;
  this.horarioVoltaDom = paramString12;
  this.pontos = paramString13;
  this.integracaoIdaIda = paramString14;
  this.integracaoIdaVolta = paramString15;
  this.integracaoVoltaIda = paramString16;
  this.integracaoVoltaVolta = paramString17;
 }

 public String getDescIda()
 {
  return this.descIda;
 }
```

```
public String getDescVolta()
{
  return this.descVolta;
}

public String getHorarioIdaDiaUtil()
{
  return this.horarioIdaDiaUtil;
}

public String getHorarioIdaDom()
{
  return this.horarioIdaDom;
}

public String getHorarioIdaSab()
{
  return this.horarioIdaSab;
}

public String getHorarioVoltaDiaUtil()
{
  return this.horarioVoltaDiaUtil;
}

public String getHorarioVoltaDom()
{
  return this.horarioVoltaDom;
}

public String getHorarioVoltaSab()
{
  return this.horarioVoltaSab;
}

public int getIdLinha()
{
  return this.idLinha;
}

public String getIntegracaoIdaIda()
{
  return this.integracaoIdaIda;
}

public String getIntegracaoIdaVolta()
{
  return this.integracaoIdaVolta;
}
```

```
  public String getIntegracaoVoltaIda()
  {
    return this.integracaoVoltaIda;
  }

  public String getIntegracaoVoltaVolta()
  {
    return this.integracaoVoltaVolta;
  }

  public String getItinerarioIda()
  {
    return this.itinerarioIda;
  }

  public String getItinerarioVolta()
  {
    return this.itinerarioVolta;
  }

  public String getNomeLinha()
  {
    return this.nomeLinha;
  }

  public String getNumLinha()
  {
    return this.numLinha;
  }

  public String getPontos()
  {
    return this.pontos;
  }
}
```