ПРИЛОЖЕНИЕ В

Исходные тексты программы.

**Листинг класса MyObject:**

package com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects;

public class MyObject {

private String name; // Название объекта

private int intId; // Идентификатор объекта

// Конструктор

public MyObject(String name, int intId) {

this.name = name;

this.intId = intId;

}

// Метод получения названия объекта

public String getName() {

return name;

}

// Метод установки имени

public void setName(String name) {

this.name = name;

}

// Метод получения идентификатора

public int getIntId() {

return intId;

}

// Метод установки идентификатора

public void setIntId(int int\_id) {

this.intId = int\_id;

}

}

**Листинг класса MainActivity:**

package com.example.vasiliy.encyclopedia\_of\_the\_sky.Activitys;

// Подключение стандартных классов

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

// Подключение спроектированых классов

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Activitys.Encyclopedia.EncyclopediaActivity;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Activitys.Game.GameMenuActivity;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.R;

/\*

Класс MainActivityреализует работу с главным экраном.

Класс реализует интерфейс OnClickListener для того,

чтобы можно было отлавливать события нажатия на кнопки

\*/

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

private Button btnToEncyc;

private Button btnToGameMenu;

private Button btnSettings;

private Button btnHelp;

/\*

Метод onCreate вызывается при создании главного экрана.

Метод осуществляет основну настройку:

Построение экрана из xml файла

Связывание кнопок с кнопками на экране.

Также определяет кнопкам обработчик событий

\*/

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

btnToEncyc = (Button) findViewById(R.id.encyc\_btn\_main);

btnToEncyc.setOnClickListener(this);

btnToGameMenu = (Button) findViewById(R.id.game\_btn\_main);

btnToGameMenu.setOnClickListener(this);

btnSettings = (Button) findViewById(R.id.settings\_btn\_main);

btnSettings.setOnClickListener(this);

btnHelp = (Button) findViewById(R.id.help\_btn\_main);

btnHelp.setOnClickListener(this);

}

/\*

Метод обработки событий.

Определяется какая кнопка нажата

\*/

@Override

public void onClick(View v) {

Intent intent = null;

switch(v.getId()) {

case R.id.encyc\_btn\_main:

intent = new Intent(MainActivity.this, EncyclopediaActivity.class);

break;

case R.id.game\_btn\_main:

intent = new Intent(MainActivity.this, GameMenuActivity.class);

break;

case R.id.settings\_btn\_main:

intent = new Intent(MainActivity.this, SettingsActivity.class);

break;

case R.id.help\_btn\_main:

intent = new Intent(MainActivity.this, HelpActivity.class);

break;

}

startActivity(intent);

}

}

package com.example.vasiliy.encyclopedia\_of\_the\_sky.Services;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.util.Log;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.ConstellationObject;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.MyObject;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.PlanetObject;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.QuestionObject;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.SkyObject;

import com.example.vasiliy.encyclopedia\_of\_the\_sky.Services.DataBaseObjects.ViewObject;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import java.util.Set;

import java.util.TreeSet;

public class SkyDataBaseImpl implements SkyDataBase{

private final String LOG\_TAG\_DB = "DataBaseQuery";

//private final Context context;

private DBHelper dbh;

private SQLiteDatabase db;

public SkyDataBaseImpl(Context context) {

//this.context = context;

dbh = new DBHelper(context);

}

private void open() {

db = dbh.getWritableDatabase();

}

private void close() {

if(dbh != null) {

dbh.close();

}

}

public void onUpdataDB() {

this.open();

dbh.onUpdata(db);

this.close();

}

public List<SkyObject> getListSkyObjects() {

this.open();

List<SkyObject> list = new ArrayList<>();

Cursor c;

c = db.query(dbh.TABLE\_NAME\_SKY\_OBJECTS, null, null, null, null, null, null);

logCursor(c);

int nameColIndex = c.getColumnIndex(dbh.TITLE\_COLUMN\_TNSO);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNSO);

int imgColIndex = c.getColumnIndex(dbh.IMG\_COLUMN\_TNSO);

if(c.moveToFirst()) {

do {

SkyObject temp = new SkyObject(c.getString(nameColIndex), c.getInt(intIdColIndex), c.getString(imgColIndex));

list.add(temp);

} while(c.moveToNext());

}

c.close();

this.close();

return list;

}

public List<SkyObject> getListConstellationSimply() {

this.open();

List<SkyObject> list = new ArrayList<>();

Cursor c;

String[] columns = {dbh.TITLE\_COLUMN\_TNC, dbh.INT\_ID\_COLUMN\_TNC};

c = db.query(dbh.TABLE\_NAME\_CONSTELLATION, columns, null, null, null, null, dbh.TITLE\_COLUMN\_TNC);

logCursor(c);

int nameColIndex = c.getColumnIndex(dbh.TITLE\_COLUMN\_TNC);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNC);

if(c.moveToFirst()) {

do {

ConstellationObject temp = new ConstellationObject(c.getString(nameColIndex), c.getInt(intIdColIndex));

list.add(temp);

} while(c.moveToNext());

}

this.close();

return list;

}

public ConstellationObject getConstellationById(int id) {

this.open();

Cursor c;

String[] columns = {

dbh.TITLE\_COLUMN\_TNC,

dbh.INT\_ID\_COLUMN\_TNC,

dbh.IMG\_COLUMN\_TNC,

dbh.TEXT\_WHERE\_FROM\_COLUMN\_TNC,

dbh.TEXT\_INF\_COLUMN\_TNC

};

String where = dbh.INT\_ID\_COLUMN\_TNC + " = ?";

String[] arg = { String.valueOf(id) };

c = db.query(dbh.TABLE\_NAME\_CONSTELLATION, columns, where, arg, null, null, null);

logCursor(c);

int titleColIndex = c.getColumnIndex(dbh.TITLE\_COLUMN\_TNC);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNC);

int imgColIndex = c.getColumnIndex(dbh.IMG\_COLUMN\_TNC);

int textInfColIndex = c.getColumnIndex(dbh.TEXT\_INF\_COLUMN\_TNC);

int textWhereColIndex = c.getColumnIndex(dbh.TEXT\_WHERE\_FROM\_COLUMN\_TNC);

c.moveToFirst();

ConstellationObject constellationObject = new ConstellationObject(

c.getString(titleColIndex),

c.getInt(intIdColIndex),

c.getString(imgColIndex),

c.getString(textInfColIndex),

c.getString(textWhereColIndex)

);

this.close();

return constellationObject;

}

public List<Integer> getConstellationIdList() {

List<Integer> list = new ArrayList<>();

this.open();

Cursor c;

String[] columns = { dbh.INT\_ID\_COLUMN\_TNC };

c = db.query(dbh.TABLE\_NAME\_CONSTELLATION, columns, null, null, null, null, dbh.TITLE\_COLUMN\_TNC);

logCursor(c);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNC);

if(c.moveToFirst()) {

do {

list.add(c.getInt(intIdColIndex));

} while(c.moveToNext());

}

this.close();

return list;

}

public List<SkyObject> getListPlanetsSimply() {

this.open();

List<SkyObject> list = new ArrayList<>();

Cursor c;

String[] columns = {dbh.TITLE\_COLUMN\_TNP, dbh.INT\_ID\_COLUMN\_TNP};

c = db.query(dbh.TABLE\_NAME\_PLANET, columns, null, null, null, null, dbh.TITLE\_COLUMN\_TNP);

logCursor(c);

int nameColIndex = c.getColumnIndex(dbh.TITLE\_COLUMN\_TNP);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNP);

if(c.moveToFirst()) {

do {

PlanetObject temp = new PlanetObject(c.getString(nameColIndex), c.getInt(intIdColIndex));

list.add(temp);

} while(c.moveToNext());

}

this.close();

return list;

}

public PlanetObject getPlanetById(int id) {

this.open();

Cursor c;

String[] columns = {

dbh.TITLE\_COLUMN\_TNP,

dbh.INT\_ID\_COLUMN\_TNP,

dbh.IMG\_COLUMN\_TNP,

dbh.MASS\_COLUMN\_TNP,

dbh.RADIUS\_COLUMN\_TNP,

dbh.DAY\_COLUMN\_TNP,

dbh.YEAR\_COLUMN\_TNP,

dbh.RADIUS\_SUN\_COLUMN\_TNP,

dbh.INFO\_COLUMN\_TNP

};

String where = dbh.INT\_ID\_COLUMN\_TNP + " = ?";

String[] arg = { String.valueOf(id) };

c = db.query(dbh.TABLE\_NAME\_PLANET, columns, where, arg, null, null, null);

logCursor(c);

c.moveToFirst();

PlanetObject planetObject = new PlanetObject(

c.getString(c.getColumnIndex(dbh.TITLE\_COLUMN\_TNP)),

c.getInt(c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.IMG\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.MASS\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.RADIUS\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.DAY\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.YEAR\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.RADIUS\_SUN\_COLUMN\_TNP)),

c.getString(c.getColumnIndex(dbh.INFO\_COLUMN\_TNP))

);

this.close();

return planetObject;

}

public List<Integer> getPlanetIdList() {

List<Integer> list = new ArrayList<>();

this.open();

Cursor c;

String[] columns = { dbh.INT\_ID\_COLUMN\_TNP };

c = db.query(dbh.TABLE\_NAME\_PLANET, columns, null, null, null, null, dbh.TITLE\_COLUMN\_TNP);

logCursor(c);

int intIdColIndex = c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNP);

if(c.moveToFirst()) {

do {

list.add(c.getInt(intIdColIndex));

} while(c.moveToNext());

}

this.close();

return list;

}

public List<MyObject> getThemes() {

this.open();

List<MyObject> list = new ArrayList<>();

Cursor c;

c = db.query(dbh.TABLE\_NAME\_THEMES, null, null, null, null, null, null);

logCursor(c);

if(c.moveToFirst()) {

do {

MyObject temp = new MyObject(c.getString(c.getColumnIndex(dbh.TITLE\_COLUMN\_TNT)), c.getInt(c.getColumnIndex(dbh.INT\_ID\_COLUMN\_TNT)));

list.add(temp);

} while(c.moveToNext());

}

c.close();

this.close();

return list;

}

public List<QuestionObject> getQuestionsOfConstellation(int amount) {

return getQuestions(amount, dbh.TABLE\_NAME\_CONSTELLATION, dbh.TITLE\_COLUMN\_TNC, dbh.INT\_ID\_COLUMN\_TNC, dbh.IMG\_COLUMN\_TNC);

}

public List<QuestionObject> getQuestionOfPlanet(int amount) {

return getQuestions(amount, dbh.TABLE\_NAME\_PLANET, dbh.TITLE\_COLUMN\_TNP, dbh.INT\_ID\_COLUMN\_TNP, dbh.IMG\_COLUMN\_TNP);

}

public ViewObject getViewObjectByName(String name) {

this.open();

Cursor c;

String[] columns = {

dbh.TITLE\_COLUMN\_TNVO,

dbh.TEXT\_COLUMN\_TNVO

};

String selection = dbh.TITLE\_COLUMN\_TNVO + " = ?";

String[] selectionArgs = { name };

c = db.query(dbh.TABLE\_NAME\_VIEW\_OBJ, columns, selection, selectionArgs, null, null, null);

logCursor(c);

c.moveToFirst();

ViewObject viewObject = new ViewObject(

c.getString(c.getColumnIndex(dbh.TITLE\_COLUMN\_TNVO)),

c.getString(c.getColumnIndex(dbh.TEXT\_COLUMN\_TNVO))

);

this.close();

return viewObject;

}

private List<QuestionObject> getQuestions(int amount, String table, String nameCol, String idCol, String imgCol) {

this.open();

List<QuestionObject> questions = new ArrayList<>();

Cursor c;

String selectCount = "select count(\*) from " + table + ";";

c = db.rawQuery(selectCount, new String[] {});

logCursor(c);

c.moveToFirst();

int count = c.getInt(c.getColumnIndex("count(\*)"));

Log.d(LOG\_TAG\_DB, String.valueOf(count));

Random random = new Random();

Set<Integer> set = new TreeSet<>();

String query1 = "select "

+ nameCol + ", "

+ idCol + ", "

+ imgCol

+ " from " + table

+ " limit ";

String query2 = ", 1;";

for(int i = 0; i < amount; ) {

int tempInt = random.nextInt(count);

if(!set.contains(tempInt)) {

set.add(tempInt);

c = db.rawQuery(query1 + String.valueOf(tempInt) + query2, new String[] {});

logCursor(c);

c.moveToFirst();

QuestionObject tempObj = new QuestionObject(

c.getString(c.getColumnIndex(nameCol)),

c.getInt(c.getColumnIndex(idCol)),

c.getString(c.getColumnIndex(imgCol))

);

Set<String> setString = new TreeSet<>();

setString.add(tempObj.getName());

tempObj.addAnswer(tempObj.getName());

for(int j = 0; j < 3; ) {

tempInt = random.nextInt(count);

c = db.rawQuery(query1 + String.valueOf(tempInt) + query2, new String[] {});

logCursor(c);

c.moveToFirst();

String answer = c.getString(c.getColumnIndex(nameCol));

if(!setString.contains(answer)) {

++j;

setString.add(answer);

tempObj.addAnswer(answer);

}

}

tempObj.mixAnswers();

questions.add(tempObj);

++i;

}

}

c.close();

this.close();

return questions;

}

private void logCursor(Cursor c) {

if (c != null) {

if (c.moveToFirst()) {

String str;

do {

str = "";

for (String cn : c.getColumnNames()) {

str = str.concat(cn + " = " + c.getString(c.getColumnIndex(cn)) + "; ");

}

Log.d(LOG\_TAG\_DB, str);

} while (c.moveToNext());

}

} else

Log.d(LOG\_TAG\_DB, "Cursor is null");

}

}