

**КОЛЛЕДЖ КОСМИЧЕСКОГО МАШИНОСТРОЕНИЯ И ТЕХНОЛОГИЙ**

**ОТЧЕТ**

по производственной практике ПП.01.01 по модулю ПМ.01  
«Разработка программных модулей программного обеспечения   
для компьютерных систем»

по специальности 09.02.03 «Программирование в компьютерных системах»

Выполнил студент гр. П1-18

Денисов М.В.

\_\_\_\_\_\_\_\_\_\_ (подпись)

Принял преподаватель

Гусятинер Л. Б.

\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_(оценка)

Королев, 2021

**Оглавление**

[Введение 3](#_Toc75745428)

[1. Общие сведения об организации 4](#_Toc75745429)

[1.1 Структура организации 4](#_Toc75745430)

[1.2 Структура отдела лаборатории 4](#_Toc75745431)

[1.3 Основные функции отдела 5](#_Toc75745432)

[1.4 Нормативные документы, которые регламентирую профессиональную деятельность 5](#_Toc75745433)

[1.5 Информационные технологии предприятия 5](#_Toc75745434)

[1.6 Программное обеспечение предприятия 5](#_Toc75745435)

[1.7 Задачи, подлежащие автоматизации 6](#_Toc75745436)

[2. Содержание выполняемых видов работ 7](#_Toc75745437)

[2.1 Разработка спецификаций отдельных компонент 7](#_Toc75745438)

[2.2 Разработка кода программного продукта на основе готовых спецификаций на уровне модуля 7](#_Toc75745439)

[2.3 Отладка программного модуля с использованием специализированных программных средств 10](#_Toc75745440)

[2.4 Тестирование программных модулей 10](#_Toc75745441)

[2.5 Оптимизация программного кода модуля 11](#_Toc75745442)

[2.6.Разработка компонентов проектной и технической документации с использованием графических языков спецификаций 13](#_Toc75745443)

[3.Выводы 14](#_Toc75745444)

[4. Дневник практики 15](#_Toc75745445)

[5.Список использованной литературы 16](#_Toc75745446)

Введение

На 3 курсе обучения в ККМТ, студентом группы П1-18 Груздевым Романом была пройдена производственная практика по модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем».

Студент получил и выполнил задание от руководителя, так же изучил организацию, для которой будет разработано данное приложение(?).

1. Общие сведения об организации
   1. Структура организации

Организационная структура университета представлена в

приложении 1.

* 1. Структура отдела лаборатории

Во главе стоит заведующий лабораторией, в его подчинении находятся студенты-практиканты.

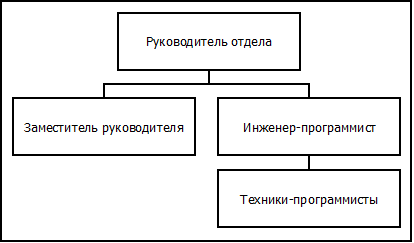


Рис 1. Организационная структура отдела

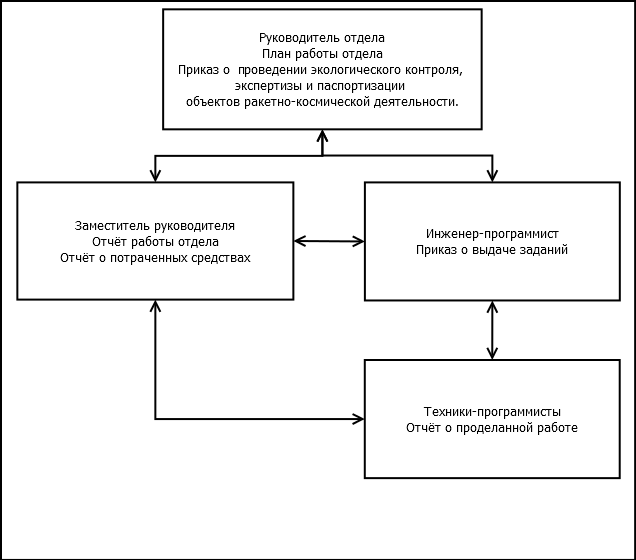


Рис 2. Информационная структура отдела

* 1. Основные функции отдела

1) Образовательная. Проведение учебных дисциплин и курсов

2) Научная. Выполнение научных исследований студентами, аспирантами, докторантами, преподавателями и сотрудниками.

* 1. Нормативные документы, которые регламентирую профессиональную деятельность
* Паспорт лаборатории
* ИОТ 23-16 (инструкция охраны труда)
* Классификация и перечень основных видов угроз информационной безопасности предприятия;
* Требования и рекомендации по обеспечению информационной безопасности предприятия;
* Положение об отделе технической защиты информации.
  1. Информационные технологии предприятия

Зав. лаборатории получает задание от Капралова, Антроповой или Трухина.

Отчёт с выполненным заданием сдаётся Антроповой, Капралову или Трухину.

* 1. Программное обеспечение предприятия
     + Windows 10
     + MS Office 2013 Pro Plus
     + 7-Zip 19.00 (x64)
     + Android Studio
     + Git version 2.24.1.2
     + Goggle Chrome
     + Notepad++ (64-bit x64)
     + Visual Studio Community 2017
     + IntelliJ IDEA Community Edition 2019.3
     + JetBrains PyCharm Community Edition 2019.2.1
  2. Задачи, подлежащие автоматизации

1. Создание систем управления рисками предприятий, органов исполнительной власти федерального и муниципального уровней.
2. Разработка систем управления проектами создания сложных технических систем.
3. Оценка рисков проектов создания сложных технических систем.
4. Содержание выполняемых видов работ

2.1 Разработка спецификаций отдельных компонент

Общее задание было разделено на 3 этапа:

1. Изучение языка программирования Java. Работа с IntelliJ IDEA.
2. Изучение платформы Android. Работа с Android Studio.
3. Написание итогового проекта.

2.2 Разработка кода программного продукта на основе готовых спецификаций на уровне модуля

Модуль отображения кнопок приведён в Листинге 1. Данный модуль осуществляет отображение текста на кнопках.

Листинг 1. Модуль отображения кнопок.

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.action\_add\_task:

final EditText taskEdit = new EditText(this);

AlertDialog dialog = new AlertDialog.Builder(this)

.setTitle("Add a new task").setMessage("Что ты хочешь сделать дальше? setView(taskEdit)

.setPositiveButton("Добавить", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

String task = String.valueOf(taskEdit.getText());

SQLiteDatabase db = taskHelper.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(TaskContract.TaskEntry.COL\_TASK\_TITLE, task);

db.insertWithOnConflict(TaskContract.TaskEntry.TABLE, null, values, SQLiteDatabase.CONFLICT\_REPLACE);

db.close();

updateUI();

}

})

.setNegativeButton("Отмена", null).create();

dialog.show();

return true;

default:

return super.onOptionsItemSelected(item);

}

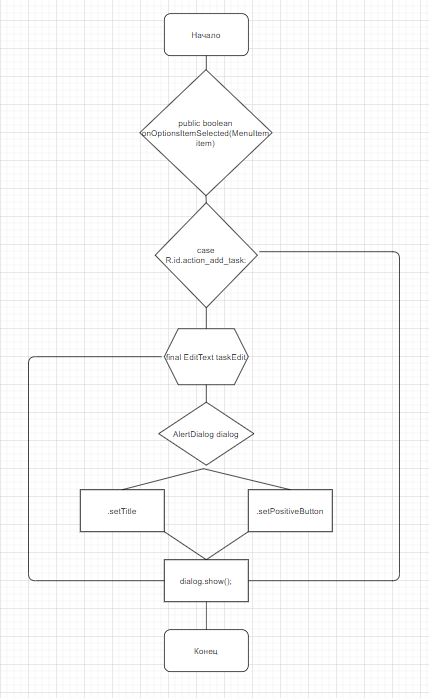


Рис 3. Модуль отображения кнопок

2.3 Отладка программного модуля с использованием специализированных программных средств

Отладка проводилась с помощью функции “Make Project” в Android Studio.

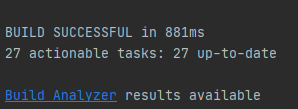


Рис. 4. Успешная сборка проекта

Также использовался Дебагер Android Studio для более лучшей проверки.

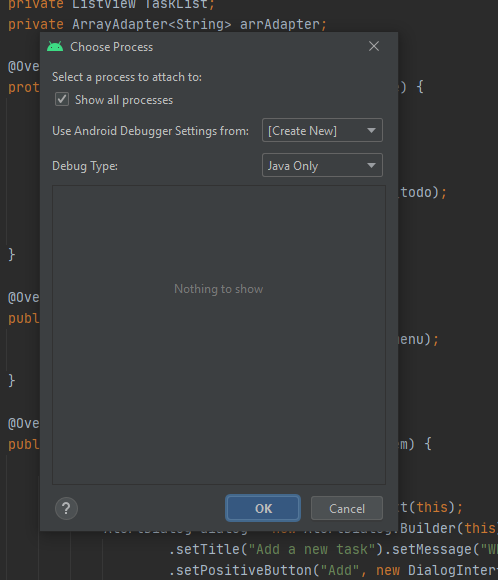


Рис 5.Пример отладчика Android Studio

2.4 Тестирование программных модулей

Тестирование приложения производиться без входных данных.

Также в программе нету выходных данных.

Выход из приложения производиться через кнопку «Назад»

2.5 Оптимизация программного кода модуля

Была оптимизирована главная функция модуля, раньше это было отдельной функцией создание новой задачи на день, после размышлений было принято объединить эти 2-ве функции в 1. Этим действием было ускоренно быстродействие программы, а также уменьшение объёма памяти.

Было:

Листинг 2. Функция добавления задач до оптимизации.

public class MyToDoApp extends Application {

private static DBToDo mDBToDo = null;

@Override

public void onCreate() {

super.onCreate();

Log.d("MyApp", "Создание DBToDo");

mDBToDo = new DBToDo(getApplicationContext());

}

Стало:

Листинг 3. Функция добавления задач после оптимизации.

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.action\_add\_task:

final EditText taskEdit = new EditText(this);

AlertDialog dialog = new AlertDialog.Builder(this)

.setTitle("Добавить новое действие").setMessage("Что ты хочешь дальше?").setView(taskEdit)

.setPositiveButton("Добавить", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

String task = String.valueOf(taskEdit.getText());

SQLiteDatabase db = taskHelper.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(TaskContract.TaskEntry.COL\_TASK\_TITLE, task);

db.insertWithOnConflict(TaskContract.TaskEntry.TABLE, null, values, SQLiteDatabase.CONFLICT\_REPLACE);

db.close();

updateUI();

}

})

2.6.Разработка компонентов проектной и технической документации с использованием графических языков спецификаций

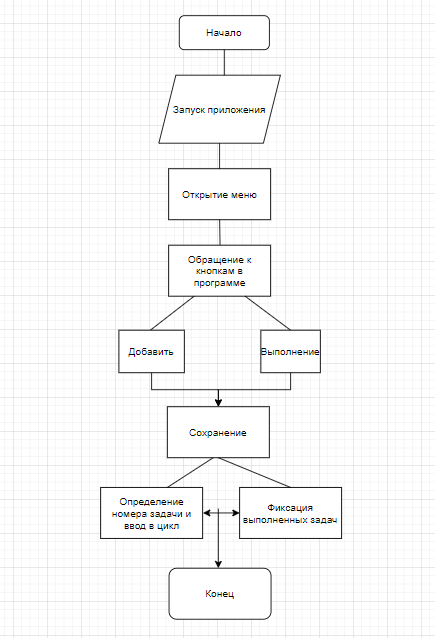


Рис 6. Общая блок-схема модуля

3.Выводы

Полученные навыки:

* + - Работа со IntelliJ IDEA Community Edition 2019.3
* Работа с базами данных в Android Studio
  + - Компиляция программы в IntelliJ IDEA Community Edition 2019.3
* Создание баз данных в Android Studio

Полученные умения:

* Разработка программ в IntelliJ IDEA Community Edition 2019.3
* Работа с файловой системой в IntelliJ IDEA Community Edition 2019.3
* Работа с файлами IntelliJ IDEA Community Edition 2019.3
* Разработка программ на IntelliJ IDEA Community Edition 2019.3

1. Дневник практики

|  |  |  |
| --- | --- | --- |
| 1. Дата | Содержание работ | Отметка о выполнении |
| 27.04-07.05 | ООП, создание объектов |  |
| 08.05-13.05 | Методы в Java |  |
| 14.05 | Интерфейсы |  |
| 15.05 | Пакеты в Java |  |
| 16.05-17.05 | Параметризованные классы (generic) |  |
| 20.05-21.05 | Обработка исключений. Try-catch |  |
| 27.05 | Функциональные интерфейсы |  |
| 28.05 | Списки в Java |  |
| 29.05 | Работа с объектами в JavaScript |  |
| 30.05-31.05 | Создание первого приложения для Android |  |
| 03.06-04.06 | Элементы приложений |  |
| 05.06-06.06 | Файл макета приложения |  |
| 07.06 | Виды компоновки элементов (Layout) |  |
| 11.06 | Элементы Android-приложений (Настройки) |  |
| 14.06-18.06 | Способы обработки нажатий кнопки в приложении |  |
| 20.06-28.06 | Составление отчета о производственной практике |  |

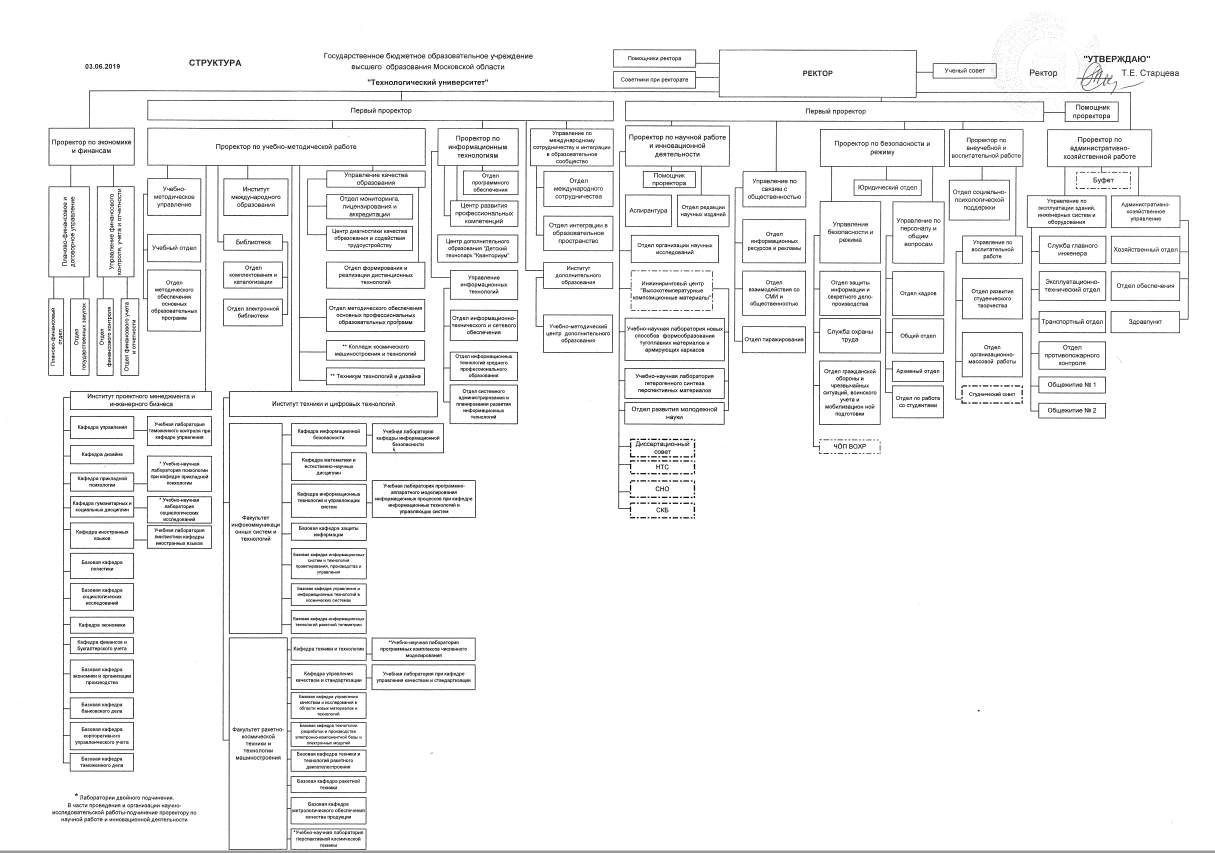
5.Список использованной литературы

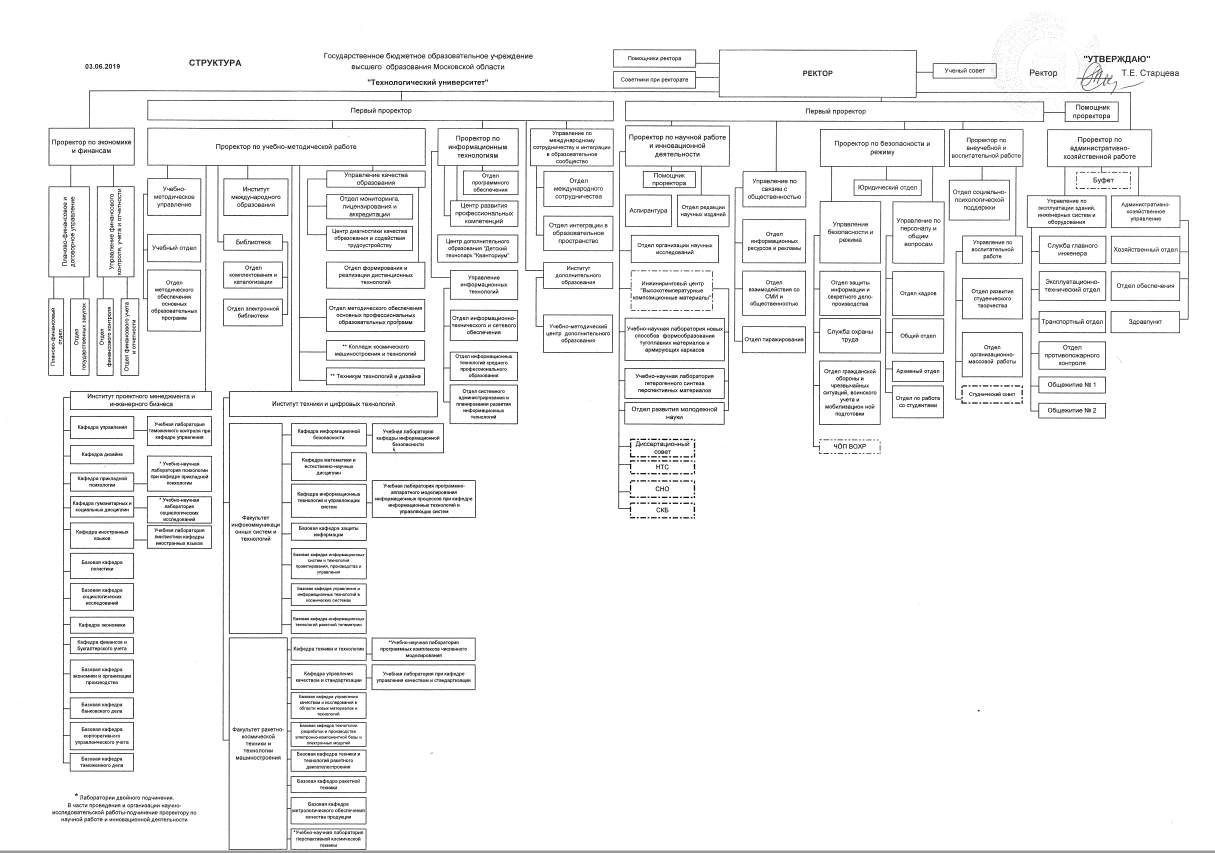
1. <https://github.com/prog-kkkmt>

2. Учение Java <http://www.java-study.ru/samouchitel.html>

3. Учение Android Studio <https://4creates.com/training/102-video-uroki-android-studio-rus.html>

4. Работа с отладчиком Android Studio <http://developer.alexanderklimov.ru/android/debug/basic.php>

Приложение 1.



Приложение 2.

package com.DataFlair.todolist;

import androidx.appcompat.app.AlertDialog;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.content.ContentValues;

import android.content.DialogInterface;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.ArrayAdapter;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.TextView;

import com.DataFlair.dataflairtodolist.task\_database.TaskContract;

import com.DataFlair.dataflairtodolist.task\_database.TaskDatabaseHelper;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

private static final String TAG = "Главная";

private TaskDatabaseHelper taskHelper;

private ListView TaskList;

private ArrayAdapter<String> arrAdapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

taskHelper = new TaskDatabaseHelper(this);

TaskList = (ListView) findViewById(R.id.list\_todo);

updateUI();

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.main\_menu, menu);

return super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.action\_add\_task:

final EditText taskEdit = new EditText(this);

AlertDialog dialog = new AlertDialog.Builder(this)

.setTitle("Добавить новое действие").setMessage("Что ты хочешь дальше?").setView(taskEdit)

.setPositiveButton("Добавить", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

String task = String.valueOf(taskEdit.getText());

SQLiteDatabase db = taskHelper.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(TaskContract.TaskEntry.COL\_TASK\_TITLE, task);

db.insertWithOnConflict(TaskContract.TaskEntry.TABLE, null, values, SQLiteDatabase.CONFLICT\_REPLACE);

db.close();

updateUI();

}

})

.setNegativeButton("Отмена", null).create();

dialog.show();

return true;

default:

return super.onOptionsItemSelected(item);

}

}

public void deleteTask(View view) {

View parent = (View) view.getParent();

TextView taskTextView = (TextView) parent.findViewById(R.id.title\_task);

String task = String.valueOf(taskTextView.getText());

SQLiteDatabase db = taskHelper.getWritableDatabase();

db.delete(TaskContract.TaskEntry.TABLE, TaskContract.TaskEntry.COL\_TASK\_TITLE + " = ?", new String[]{task});

db.close();

updateUI();

}

private void updateUI() {

ArrayList<String> taskList = new ArrayList<>();

SQLiteDatabase db = taskHelper.getReadableDatabase();

Cursor cursor = db.query(TaskContract.TaskEntry.TABLE,

new String[]{TaskContract.TaskEntry.\_ID, TaskContract.TaskEntry.COL\_TASK\_TITLE},

null, null, null, null, null);

while (cursor.moveToNext()) {

int idx = cursor.getColumnIndex(TaskContract.TaskEntry.COL\_TASK\_TITLE);

taskList.add(cursor.getString(idx));

}

if (arrAdapter == null) {

arrAdapter = new ArrayAdapter<>(this, R.layout.todo\_task, R.id.title\_task, taskList);

TaskList.setAdapter(arrAdapter);

} else {

arrAdapter.clear();

arrAdapter.addAll(taskList);

arrAdapter.notifyDataSetChanged();

}

cursor.close();

db.close();

}

}

Приложение 3.

package com.DataFlair.dataflairtodolist.task\_database;

import android.provider.BaseColumns;

public class TaskContract {

public static final String DB\_NAME = "com.aziflaj.todolist.db";

public static final int DB\_VERSION = 1;

public class TaskEntry implements BaseColumns {

public static final String TABLE = "tasks";

public static final String COL\_TASK\_TITLE = "title";

}

}

Приложение 4.

package com.DataFlair.dataflairtodolist.task\_database;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

public class TaskDatabaseHelper extends SQLiteOpenHelper {

public TaskDatabaseHelper(Context context) {

super(context, TaskContract.DB\_NAME, null, TaskContract.DB\_VERSION);

}

@Override

public void onCreate(SQLiteDatabase db) {

String createTable = "CREATE TABLE " + TaskContract.TaskEntry.TABLE + " ( " +

TaskContract.TaskEntry.\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

TaskContract.TaskEntry.COL\_TASK\_TITLE + " TEXT NOT NULL);";

db.execSQL(createTable);

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

db.execSQL("DROP TABLE IF EXISTS " + TaskContract.TaskEntry.TABLE);

onCreate(db);

}

}