**Пословицы о котах**

Напишем приложение, которое будет хранить пословицы о котах в базе данных SQLite и выводить их на экран в случайном порядке.

У пользователя будет возможность внести в базу новую пословицу. Таким образом, он сможет пополнять свою базу. А при нажатии на вторую кнопку будет выводиться пословица в случайном порядке.

В первом месяце обучения мы уже знакомились с основами работы с базой данных на SQLite и данный материал предназначен для закрепления навыков.

Создадим новый проект **RandomProverbs**.

Начнём с класса базы данных ProverbDB.

package ru.alexanderklimov.randomproverbs;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

import android.util.Log;

public class ProverbDB extends SQLiteOpenHelper {

// константы для конструктора

// имя файла для базы данных

private static final String DATABASE\_NAME = "proverbs.db";

// номер версии базы данных

private static final int DATABASE\_VERSION = 1;

// имя таблицы

public static final String TABLE\_NAME = "tableRandomProverbs";

// Колонка

public static final String KEY\_ROWID = "\_id";

// Вторая колонка

public static final String KEY\_PROVERB = "Proverb";

// для LogCat

private static final String TAG = "RandomProverb1098";

// Строка для создания таблицы в базе данных

private static final String DATABASE\_CREATE = "CREATE TABLE " + TABLE\_NAME

+ " (" + KEY\_ROWID + " INTEGER PRIMARY KEY AUTOINCREMENT,"

+ KEY\_PROVERB + " VARCHAR(255));";

// Строка для удаления таблицы из базы данных

private static final String DELETE\_ENTRIES = "DROP TABLE IF EXISTS "

+ TABLE\_NAME;

private static ProverbDB dbInstance;

// Конструктор

public ProverbDB(Context context) {

// TODO Auto-generated constructor stub

super(context, DATABASE\_NAME, null, DATABASE\_VERSION);

}

public static ProverbDB getInstance(Context context) {

if (dbInstance == null)

dbInstance = new ProverbDB(context.getApplicationContext());

return dbInstance;

}

@Override

public void onCreate(SQLiteDatabase db) {

// TODO Auto-generated method stub

db.execSQL(DATABASE\_CREATE);

}

@Override

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

// TODO Auto-generated method stub

Log.w(TAG, "Обновление базы данных с версии " + oldVersion

+ " до версии " + newVersion

+ ", которое удалит все старые данные");

// Удаляем предыдущую таблицу при апгрейде

db.execSQL(DELETE\_ENTRIES);

// Создаём новый экземпляр таблицы

onCreate(db);

}

}

Комментарии в коде поясняют логику программы, там ничего сложного и всё знакомо.

Перейдем к основной активности. Нам понадобится текстовое поле для ввода новой пословицы, кнопки для ввода и вывода данных и текстовая метка для вывода пословицы.

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="vertical" >

<TextView

android:id="@+id/tvProverb"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Добавить пословицу" />

<EditText

android:id="@+id/editProverb"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content" />

<Button

android:id="@+id/buttonInsert"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Добавить" />

<Button

android:id="@+id/buttonGetRandom"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Показать" />

<TextView

android:id="@+id/tvShow"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:textColor="#0000ff" />

</LinearLayout>

Теперь напишем код:

package ru.alexanderklimov.randomproverbs;

import java.util.Random;

import android.os.Bundle;

import android.app.Activity;

import android.content.ContentValues;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.view.Menu;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class MainActivity extends Activity {

private ProverbDB sqh;

private SQLiteDatabase db;

EditText editProverb;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Button butAdd = (Button) findViewById(R.id.buttonInsert);

Button butGet = (Button) findViewById(R.id.buttonGetRandom);

// Регистрируем приемник OnClick

butAdd.setOnClickListener(mClickListener);

butGet.setOnClickListener(mClickListener);

// Инициализируем наш класс-обёртку

sqh = new ProverbDB(this);

}

@Override

protected void onDestroy() {

// TODO Auto-generated method stub

// закрываем соединения с базой данных

sqh.close();

super.onDestroy();

}

// обработчик нажатия кнопок

private OnClickListener mClickListener = new OnClickListener() {

public void onClick(View v) {

switch (v.getId()) {

case R.id.buttonInsert:

// База нам нужна для записи

db = sqh.getWritableDatabase();

try {

editProverb = (EditText) findViewById(R.id.editProverb);

insertProverb(editProverb.getText().toString());

Context context = getApplicationContext();

CharSequence text = "Пословица '" + editProverb.getText()

+ "' успешно добавлена!\nВсего пословиц: "

+ getAllEntries();

int duration = Toast.LENGTH\_LONG;

Toast toast = Toast.makeText(context, text, duration);

toast.show();

editProverb.setText("");

} catch (Exception ex) {

CharSequence text = ex.toString();

Toast.makeText(getApplicationContext(), text, Toast.LENGTH\_LONG).show();

}

// закрываем соединения с базой данных

db.close();

break;

case R.id.buttonGetRandom:

db = sqh.getReadableDatabase();

try {

String proverb = "";

proverb = getRandomEntry();

TextView tvShowProverb = (TextView)findViewById(R.id.tvShow);

tvShowProverb.setText(proverb);

} catch (Exception ex) {

Toast.makeText(getApplicationContext(), ex.toString(),

Toast.LENGTH\_LONG).show();

}

// закрываем соединения с базой данных

db.close();

}

}

};

// вставляем новую запись в базу данных

public long insertProverb(String proverb) {

ContentValues initialValues = new ContentValues();

initialValues.put(ProverbDB.KEY\_PROVERB, proverb);

return db.insert(ProverbDB.TABLE\_NAME, null, initialValues);

}

// получим общее количество пословиц в базе данных

public int getAllEntries() {

String query = "SELECT COUNT(\*) FROM tableRandomProverbs";

Cursor cursor = db.rawQuery(query, null);

int count = 0;

while (cursor.moveToNext()) {

count = cursor.getInt(0);

return count;

}

cursor.close();

return count;

}

public String getRandomEntry() {

// подключаем генератор случайных числе

Random random = new Random();

// задаем диапазон

int rand = random.nextInt(getAllEntries() + 1);

if (rand == 0)

++rand;

Cursor cursor = db.rawQuery("SELECT Proverb FROM " + ProverbDB.TABLE\_NAME

+ " WHERE \_id = " + rand, null);

if (cursor.moveToFirst()) {

return cursor.getString(0);

}

cursor.close();

return cursor.getString(0);

}

}

Метод **getAllEntries()** будет выполнять запрос на количество введенных в БД пословиц. Это значение понадобится нам для генератора случайных чисел, чтобы выбрать нужный диапазон.

Метод **getRandomEntry()** служит для возвращения случайного результата, используя число записей в базе дынных, полученных через метод **getAllEntries()**. В SQL-операторе **SELECT**мы выбираем пословицу под номером по заданному случайному идентификатору ID, WHERE \_id = rand.

Вот приложение и готово. Если у вас возникнут трудности с добавлением новых пословиц о котах, то скачайте программу [Справочник про котов](https://play.google.com/store/apps/details?id=ru.alexanderklimov.crib). Там есть целый раздел на данную тему.

 