**Использование существующей БД SQLite**

Кроме создания новой базы данных мы также можем использовать уже существующую. Это может быть более предпочтительно, так как в этом случае база данных приложения уже будет содержать всю необходимую информацию.

Возьмем проект, созданный в предыдущих темах. Для начала создадим базу данных SQLite. В этом нам может помочь такой инструмент как **Sqlitebrowser**. Он бесплатный и доступен для различных операционных систем по адресу <http://sqlitebrowser.org/>. Хотя можно использовать и другие способы для создания начальной БД.

После создания таблиц, добавим в проект в Android Studio папку **assets**, а в папку assets - только что созданную базу данных. Для этого перейдем к полному опеределению проекта, нажмем на папку main правой кнопкой мыши и в меню выберем **New -> Directory**:

Назовем добавляемую папку assets и затем скопируем в нее нашу базу данных:

В моем случае база данных называется "cityinfo.db".

Изменим код DatabaseHelper следующим образом:

package com.example.eugene.sqliteapp;

import android.database.SQLException;

import android.database.sqlite.SQLiteOpenHelper;

import android.database.sqlite.SQLiteDatabase;

import android.content.Context;

import android.util.Log;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

class DatabaseHelper extends SQLiteOpenHelper {

    private static String DB\_PATH; // полный путь к базе данных

    private static String DB\_NAME = "cityinfo.db";

    private static final int SCHEMA = 1; // версия базы данных

    static final String TABLE = "users"; // название таблицы в бд

    // названия столбцов

    static final String COLUMN\_ID = "\_id";

    static final String COLUMN\_NAME = "name";

    static final String COLUMN\_YEAR = "year";

    private Context myContext;

    DatabaseHelper(Context context) {

        super(context, DB\_NAME, null, SCHEMA);

        this.myContext=context;

        DB\_PATH =context.getFilesDir().getPath() + DB\_NAME;

    }

    @Override

    public void onCreate(SQLiteDatabase db) {

    }

    @Override

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

    }

    void create\_db(){

        InputStream myInput = null;

        OutputStream myOutput = null;

        try {

            File file = new File(DB\_PATH);

            if (!file.exists()) {

                this.getReadableDatabase();

                //получаем локальную бд как поток

                myInput = myContext.getAssets().open(DB\_NAME);

                // Путь к новой бд

                String outFileName = DB\_PATH;

                // Открываем пустую бд

                myOutput = new FileOutputStream(outFileName);

                // побайтово копируем данные

                byte[] buffer = new byte[1024];

                int length;

                while ((length = myInput.read(buffer)) > 0) {

                    myOutput.write(buffer, 0, length);

                }

                myOutput.flush();

                myOutput.close();

                myInput.close();

            }

        }

        catch(IOException ex){

            Log.d("DatabaseHelper", ex.getMessage());

        }

    }

    public SQLiteDatabase open()throws SQLException {

        return SQLiteDatabase.openDatabase(DB\_PATH, null, SQLiteDatabase.OPEN\_READWRITE);

    }

}

По умолчанию база данных будет размещаться во внешнем хранилище, выделяемом для приложения в папке **/data/data/[название\_пакета]/databases/**, и чтобы получить полный путь к базе данных в конструкторе используется выражение:

|  |  |
| --- | --- |
| 1 | DB\_PATH =context.getFilesDir().getPath() + DB\_NAME; |

Метод onCreate() нам не нужен, так как нам не требуется создание встроенной базы данных. Зато здесь определен дополнительный методcreate\_db(), цель которого копирование базы данных из папки assets в то место, которое указано в переменной DB\_PATH.

Кроме этого здесь также определен метод открытия базы данных open() с помощью метода SQLiteDatabase.openDatabase()

Новый способ организации подключения изменит использование DatabaseHelper в activity. Так, обновим класс **MainActivity**:

package com.example.eugene.sqliteapp;

import android.content.Intent;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ListView;

import android.widget.SimpleCursorAdapter;

public class MainActivity extends AppCompatActivity {

    ListView userList;

    DatabaseHelper databaseHelper;

    SQLiteDatabase db;

    Cursor userCursor;

    SimpleCursorAdapter userAdapter;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        userList = (ListView)findViewById(R.id.list);

        userList.setOnItemClickListener(new AdapterView.OnItemClickListener() {

            @Override

            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

                Intent intent = new Intent(getApplicationContext(), UserActivity.class);

                intent.putExtra("id", id);

                startActivity(intent);

            }

        });

        databaseHelper = new DatabaseHelper(getApplicationContext());

        // создаем базу данных

        databaseHelper.create\_db();

    }

    @Override

    public void onResume() {

        super.onResume();

        // открываем подключение

        db = databaseHelper.open();

        //получаем данные из бд в виде курсора

        userCursor =  db.rawQuery("select \* from "+ DatabaseHelper.TABLE, null);

        // определяем, какие столбцы из курсора будут выводиться в ListView

        String[] headers = new String[] {DatabaseHelper.COLUMN\_NAME, DatabaseHelper.COLUMN\_YEAR};

        // создаем адаптер, передаем в него курсор

        userAdapter = new SimpleCursorAdapter(this, android.R.layout.two\_line\_list\_item,

                userCursor, headers, new int[]{android.R.id.text1, android.R.id.text2}, 0);

        userList.setAdapter(userAdapter);

    }

    // по нажатию на кнопку запускаем UserActivity для добавления данных

    public void add(View view){

        Intent intent = new Intent(this, UserActivity.class);

        startActivity(intent);

    }

    @Override

    public void onDestroy(){

        super.onDestroy();

        // Закрываем подключение и курсор

        db.close();

        userCursor.close();

    }

}

И также изменим класс **UserActivity**:

package com.example.eugene.sqliteapp;

import android.content.ContentValues;

import android.content.Intent;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

public class UserActivity extends AppCompatActivity {

    EditText nameBox;

    EditText yearBox;

    Button delButton;

    Button saveButton;

    DatabaseHelper sqlHelper;

    SQLiteDatabase db;

    Cursor userCursor;

    long userId=0;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_user);

        nameBox = (EditText) findViewById(R.id.name);

        yearBox = (EditText) findViewById(R.id.year);

        delButton = (Button) findViewById(R.id.deleteButton);

        saveButton = (Button) findViewById(R.id.saveButton);

        sqlHelper = new DatabaseHelper(this);

        db = sqlHelper.open();

        Bundle extras = getIntent().getExtras();

        if (extras != null) {

            userId = extras.getLong("id");

        }

        // если 0, то добавление

        if (userId > 0) {

            // получаем элемент по id из бд

            userCursor = db.rawQuery("select \* from " + DatabaseHelper.TABLE + " where " +

                    DatabaseHelper.COLUMN\_ID + "=?", new String[]{String.valueOf(userId)});

            userCursor.moveToFirst();

            nameBox.setText(userCursor.getString(1));

            yearBox.setText(String.valueOf(userCursor.getInt(2)));

            userCursor.close();

        } else {

            // скрываем кнопку удаления

            delButton.setVisibility(View.GONE);

        }

    }

    public void save(View view){

        ContentValues cv = new ContentValues();

        cv.put(DatabaseHelper.COLUMN\_NAME, nameBox.getText().toString());

        cv.put(DatabaseHelper.COLUMN\_YEAR, Integer.parseInt(yearBox.getText().toString()));

        if (userId > 0) {

            db.update(DatabaseHelper.TABLE, cv, DatabaseHelper.COLUMN\_ID + "=" + String.valueOf(userId), null);

        } else {

            db.insert(DatabaseHelper.TABLE, null, cv);

        }

        goHome();

    }

    public void delete(View view){

        db.delete(DatabaseHelper.TABLE, "\_id = ?", new String[]{String.valueOf(userId)});

        goHome();

    }

    private void goHome(){

        // закрываем подключение

        db.close();

        // переход к главной activity

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

        intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP | Intent.FLAG\_ACTIVITY\_SINGLE\_TOP);

        startActivity(intent);

    }

}

Вся остальная работа с данными будет той же, чтобы и в прошлых темах:

