**Добавление удаление и обновление данных в SQLite**

Продолжим работу с проектом из прошлой темы, где мы получаем данные. Теперь добавим в него стандартную CRUD-логику (создание, обновление, удаление).

Чтобы не нагромождать форму с главной activity, все остальные действия по работе с данными будут происходить на другом экране. Добавим в проект в папку *res/layout* новый файл *fragment\_user.xml*:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      android:orientation="vertical"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent">      <EditText          android:id="@+id/name"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:hint="Введите имя"/>      <EditText          android:id="@+id/year"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:hint="Введите год рождения"/>      <LinearLayout          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:orientation="horizontal"          android:layout\_weight="2">            <Button              android:id="@+id/save"              android:layout\_width="wrap\_content"              android:layout\_weight="1"              android:layout\_height="wrap\_content"              android:text="Сохранить" />          <Button              android:id="@+id/delete"              android:layout\_width="wrap\_content"              android:layout\_weight="1"              android:layout\_height="wrap\_content"              android:text="Удалить" />            </LinearLayout>    </LinearLayout> |

Это будет универсальная форма для добавления/обновления/удаления данных.

И также добавим новую activity. Назовем ее UserActivity:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129 | package com.example.eugene.sqlitedbapp;    import android.content.ContentValues;  import android.database.Cursor;  import android.database.sqlite.SQLiteDatabase;  import android.support.v7.app.ActionBarActivity;  import android.support.v4.app.Fragment;  import android.os.Bundle;  import android.view.LayoutInflater;  import android.view.View;  import android.view.ViewGroup;  import android.widget.Button;  import android.widget.EditText;  import android.content.Intent;    public class UserActivity extends ActionBarActivity {        @Override      protected void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_user);            long userId=0;          Bundle extras = getIntent().getExtras();          if (extras != null) {              userId = extras.getLong("id");          }            if (savedInstanceState == null) {              getSupportFragmentManager().beginTransaction()                      .add(R.id.container, PlaceholderFragment.newInstance(userId))                      .commit();          }      }        public static class PlaceholderFragment extends Fragment {            EditText nameBox;          EditText yearBox;          Button delButton;          Button saveButton;            DatabaseHelper sqlHelper;          SQLiteDatabase db;          Cursor userCursor;            public static PlaceholderFragment newInstance(long id) {              PlaceholderFragment fragment = new PlaceholderFragment();              Bundle args=new Bundle();              args.putLong("id", id);              fragment.setArguments(args);              return fragment;          }            @Override          public void onCreate(Bundle savedInstanceState) {              super.onCreate(savedInstanceState);              setRetainInstance(true);                sqlHelper = new DatabaseHelper(getActivity());          }          public PlaceholderFragment() {          }            @Override          public View onCreateView(LayoutInflater inflater, ViewGroup container,                                   Bundle savedInstanceState) {              View rootView = inflater.inflate(R.layout.fragment\_user, container, false);              nameBox = (EditText) rootView.findViewById(R.id.name);              yearBox = (EditText) rootView.findViewById(R.id.year);              delButton = (Button) rootView.findViewById(R.id.delete);              saveButton = (Button) rootView.findViewById(R.id.save);                final long id = getArguments() != null ? getArguments().getLong("id") : 0;                db = sqlHelper.getWritableDatabase();              // кнопка удаления              delButton.setOnClickListener(new View.OnClickListener() {                  @Override                  public void onClick(View v) {                        db.delete(DatabaseHelper.TABLE, "\_id = ?", new String[]{String.valueOf(id)});                      goHome();                  }              });                // кнопка сохранения              saveButton.setOnClickListener(new View.OnClickListener() {                  @Override                  public void onClick(View v) {                      ContentValues cv = new ContentValues();                      cv.put(DatabaseHelper.COLUMN\_NAME, nameBox.getText().toString());                      cv.put(DatabaseHelper.COLUMN\_YEAR, Integer.parseInt(yearBox.getText().toString()));                        if (id > 0) {                          db.update(DatabaseHelper.TABLE, cv, DatabaseHelper.COLUMN\_ID + "=" + String.valueOf(id), null);                      } else {                          db.insert(DatabaseHelper.TABLE, null, cv);                      }                      goHome();                  }              });                // если 0, то добавление              if (id > 0) {                  // получаем элемент по id из бд                  userCursor = db.rawQuery("select \* from " + DatabaseHelper.TABLE + " where " +                          DatabaseHelper.COLUMN\_ID + "=?", new String[]{String.valueOf(id)});                  userCursor.moveToFirst();                  nameBox.setText(userCursor.getString(1));                  yearBox.setText(String.valueOf(userCursor.getInt(2)));                  userCursor.close();              } else {                  // скрываем кнопку удаления                  delButton.setVisibility(View.GONE);               }              return rootView;          }            public void goHome(){              // закрываем подключение              db.close();              // переход к главной activity              Intent intent = new Intent(getActivity(), MainActivity.class);              intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP | Intent.FLAG\_ACTIVITY\_SINGLE\_TOP);              startActivity(intent);          }      }  } |

При обновлении или удалении объекта из списка из главной activity в UserActivity будет передаваться id объекта:

|  |  |
| --- | --- |
| 1  2  3  4  5 | long userId=0;  Bundle extras = getIntent().getExtras();  if (extras != null) {      userId = extras.getLong("id");  } |

Для упрощения работы с данными вся логика вынесена в отдельный фрагмент PlaceholderFragment. В методе onCreate этот фрагмент добавляется в визуальный интерфейс activity:

|  |  |
| --- | --- |
| 1  2  3 | getSupportFragmentManager().beginTransaction()              .add(R.id.container, PlaceholderFragment.newInstance(userId))              .commit(); |

При создании фрагмента ему будет передаваться полученный id редактируемого или удаляемого объекта. Уже в самом фрагменте этот id сохраняется в аргументах:

|  |  |
| --- | --- |
| 1  2  3 | Bundle args=new Bundle();  args.putLong("id", id);  fragment.setArguments(args); |

В методе onCreateView() также создаем объекты для работы с базой данных и получаем из аргументов id:

|  |  |
| --- | --- |
| 1 | final long id = getArguments() != null ? getArguments().getLong(id") : 0; |

Если из MainActivity не было передано id, то устанавливаем его значение 0, следовательно, у нас будет добавление, а не редактирование/удаление

Если id определен, то получаем по нему из базы данных объект для редактирования/удаления:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | if (id < 0) {        userCursor = db.rawQuery("select \* from " + DatabaseHelper.TABLE + " where " +              DatabaseHelper.COLUMN\_ID + "=?", new String[]{String.valueOf(id)});      userCursor.moveToFirst();      nameBox.setText(userCursor.getString(1));      yearBox.setText(String.valueOf(userCursor.getInt(2)));      userCursor.close();  } |

Иначе просто скрываем кнопку удаления.

Для выполнения операций по вставке, обновлению и удалению данных SQLiteDatabase имеет методыinsert(), update() и delete(). Эти методы вызываются в обработчиках кнопок:

|  |  |
| --- | --- |
| 1 | db.delete(DatabaseHelper.TABLE, "\_id = ?", new String[]{String.valueOf(id)}); |

В метод delete() передается название таблицы, а также столбец, по которому происходит удаление, и его значение. В качестве критерия можно выбрать несколько столбцов, поэтому третьим параметром идет массив. Знак вопроса ? обозначает параметр, вместо которого подставляется значение из третьего параметра.

Для добавления или обновления нам надо создать объект **ContentValues**. Его метод put() позволяет добавить набор значений:

|  |  |
| --- | --- |
| 1  2  3 | ContentValues cv = new ContentValues();  cv.put(DatabaseHelper.COLUMN\_NAME, nameBox.getText().toString());  cv.put(DatabaseHelper.COLUMN\_YEAR, Integer.parseInt(yearBox.getText().toString())); |

При обновлении в метод update() передается название таблицы, объект ContentValues и критерий, по которому происходит обновление (в данном случае столбец id):

|  |  |
| --- | --- |
| 1 | db.update(DatabaseHelper.TABLE, cv, DatabaseHelper.COLUMN\_ID + "=" + String.valueOf(id), null); |

Метод insert() принимает название таблицы, объект ContentValues с добавляемыми значениями. Второй параметр является необязательным: он передает столбец, в который надо добавить значение NULL:

|  |  |
| --- | --- |
| 1 | db.insert(DatabaseHelper.TABLE, null, cv); |

Вместо этих методов, как в прошлой теме, можно использовать метод execSQL() с точным указанием выполняемого sql-выражения. В то же время методы delete/insert/update имеют преимущество - они возвращают id измененной записи, по которому мы можем узнать об успешности операции, или -1 в случае неудачной операции:

|  |  |
| --- | --- |
| 1  2  3  4  5 | long result = db.insert(DatabaseHelper.TABLE, null, cv);  if(result>0){        // действия  } |

После каждой операции выполняется метод goHome(), который возвращает на главную activity.

Изменим файл интерфейса *activity\_user.xml* для UserActivity:

|  |  |
| --- | --- |
| 1  2  3  4 | <FrameLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      xmlns:tools="<http://schemas.android.com/tools>" android:id="@+id/container"      android:layout\_width="match\_parent" android:layout\_height="match\_parent"      tools:context="com.example.eugene.sqlitedbapp.UserActivity" tools:ignore="MergeRootFrame" /> |

Теперь файл *activity\_user.xml* содержит элемент, в который будет вставляться выше определенный фрагмент.

После этого нам надо исправить код MainActivity, чтобы она инициировала выполнение кода в UserActivity. Для этого в MainActivity изменим метод onCreate():

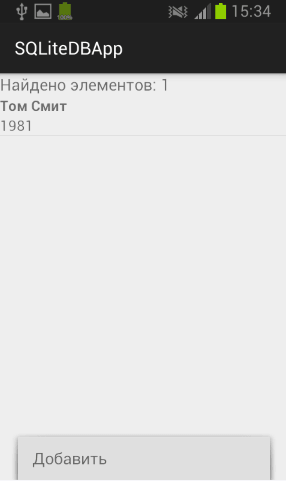
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | @Override  protected void onCreate(Bundle savedInstanceState) {      super.onCreate(savedInstanceState);      setContentView(R.layout.activity\_main);        header = (TextView)findViewById(R.id.header);        mList = (ListView)findViewById(R.id.list);      mList.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);          }      });      sqlHelper = new DatabaseHelper(getApplicationContext());  } |

В обработчике нажатия списка будет запускаться UserActivity с передачей ей id выбранной записи.

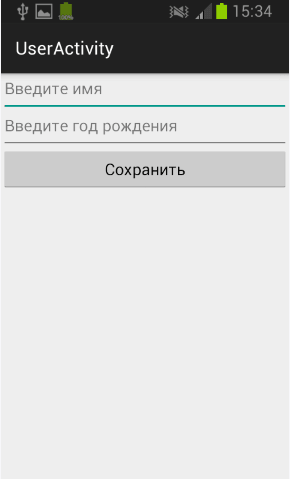
Для добавления новых элементов, когда у нас еще нет никакого id, изменим код обработки меню в MainActivity:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | @Override  public boolean onOptionsItemSelected(MenuItem item) {        int id = item.getItemId();        if (id == R.id.action\_settings) {            Intent intent = new Intent(getApplicationContext(), UserActivity.class);          startActivity(intent);          return true;      }      return super.onOptionsItemSelected(item);  } |

Запустим приложение и выберем единственный пункт меню, который должен отправлять на UserActivity:



И мы попадем на форму добавления, где сможем добавить новый объект:



При нажатии в MainActivity на элемент списка этот элемент попадет на UserActivity, где его можно будет удалить или подредактировать:

