МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**ОТЧЕТ** **ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

Дисциплина: **«**Программирование для мобильных устройств**»**

Тема: **«**Использование управляющих элементов в пользовательском интерфейсе. Работа с базами данных в Android**»**

ВЫПОЛНИЛ студент группы 16-ИТ-1

Дубовский Н.В.

ПРОВЕРИЛ Лукьянов А.О.

Полоцк 2019 г.

**Цель работы**

Научиться использовать в интерфейсе пользователя различные управляющие элементы: кнопки с изображениями, радиокнопки, чек боксы и другие. Научиться взаимодействовать с базой данных в Android приложении.

**Теоретическая часть**

**Использование графической кнопки**

Для использования изображения вместо текста на кнопке потребуются три изображения для трех состояний кнопки: обычное, выбранное («в фокусе») и нажатое. Все эти три изображения с соответствующими состояниями описываются в одном XML файле, который используется для создания такой кнопки.

**Использование виджета ToggleButton**

Данный виджет хорошо подходит в качестве альтернативы радиокнопкам и чек боксам, когда требуется переключаться между двумя взаимоисключающими состояниями, например, Включено/Выключено.

**Использование виджета RadioButton**

Радиокнопки используются для выбора между различными взаимоисключающими вариантами. Для создания группы радиокнопок используется элемент RadioGroup, внутри которого располагаются элементы RadioButton.

**Использование виджета EditText**

Виджет EditText используется для ввода текста пользователем. Установленный для этого виджета обработчик нажатий на кнопки будет показывать введенный текст с помощью Toast.

**Работа с БД**

Механизм работы с базами данных в Android позволяет хранить и обрабатывать структурированную информацию. Любое приложение может создавать свои собственные базы данных, над которыми оно будет иметь полный контроль.

В Android используется библиотека SQLite, представляющую из себя реляционную СУБД, обладающую следующими характерными особенностями: свободно распространяемая (open source), поддерживающая стандартный язык запросов и транзакции, легковесная, одноуровневая (встраиваемая), отказоустойчивая.

**Курсоры (Cursor) и ContentValues**

Запросы к СУБД возвращают объекты типа Cursor. Для экономии ресурсов используется подход, когда при извлечении данных не возвращаются копии их значений из СУБД, а создается Cursor, обеспечивающий навигацию и доступ к запрошенному набору исходных данных. Методы объекта Cursor предоставляют различные возможности навигации, назначение которых, как правило, понятно из названия:

• moveToFirst

• moveToNext

• moveToPrevious

• getCount

• getColumnIndexOrThrow

• getColumnName

• getColumnNames

• moveToPosition

• getPosition

**Особенности работы с БД в Android**

При работе с базами данных в Android следует избегать хранения BLOB'ов с таблицах из-за резкого падения эффективности работы.

Как показано в примере адаптера БД, для каждой таблицы рекомендуется создавать автоикрементное поле **\_id**, которое будет уникальным индексом для строк. Если же планируется делегировать доступ к БД с помощью контент-провайдеров, такое поле является обязательным.

**Использование SimpleCursorAdapter**

SimpleCursorAdapter позволяет привязать курсор к ListView. используя описание разметки для отображения строк и полей.

**Выполнение лабараторной работы**

В данной лабораторной работе выполнено и скомпилировано приложение, в главной активности будет реализованы все нужные графические элементы, также будут реализованы активности для работы с БД один с адаптером, другой без.

*<?*xml version="1.0" encoding="utf-8"*?>*<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <LinearLayout  
 android:layout\_width="368dp"  
 android:layout\_height="499dp"  
 android:orientation="vertical"  
 tools:layout\_editor\_absoluteX="8dp"  
 tools:layout\_editor\_absoluteY="8dp"  
 tools:ignore="MissingConstraints">  
  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/smile\_button"  
 android:onClick="onButtonClicked"  
 android:padding="10dp"  
 tools:ignore="MissingConstraints"  
 tools:layout\_editor\_absoluteX="8dp"  
 tools:layout\_editor\_absoluteY="225dp" />  
 <Button  
 android:id="@+id/button1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/smile\_button1"  
 android:onClick="onButtonClicked1"  
 android:padding="10dp"  
 tools:ignore="MissingConstraints,OnClick"  
 tools:layout\_editor\_absoluteX="8dp"  
 tools:layout\_editor\_absoluteY="225dp" />  
 <CheckBox  
 android:id="@+id/checkbox"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="onCheckboxClicked"  
 android:text="Выбери меня" />  
 <ToggleButton android:id="@+id/togglebutton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textOn="Звонок включен"  
 android:textOff="Звонок выключен"  
 android:onClick="onToggleClicked"/>  
 <RadioGroup  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical" >  
 <RadioButton  
 android:id="@+id/radio\_dog"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="onRadioButtonClicked"  
 android:text="Собака" />  
 <RadioButton  
 android:id="@+id/radio\_cat"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="onRadioButtonClicked"  
 android:text="Кошка" />  
 <RadioButton  
 android:id="@+id/radio\_rabbit"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="onRadioButtonClicked"  
 android:text="Кролик" />  
 </RadioGroup>  
 <Button  
 android:id="@+id/button2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Очистить"  
 android:onClick="clean"  
 />  
 <EditText  
 android:id="@+id/user\_name"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Введите имя"/>  
 </LinearLayout>  
  
</android.support.constraint.ConstraintLayout>

Листинг 1 – activity\_main

*<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".DBActivity">  
  
 <ListView  
 android:id="@+id/lv"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
 </ListView>  
</android.support.constraint.ConstraintLayout>*

Листинг 2 – activity\_db

*<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".AdapterDBActivity">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
 <EditText  
 android:id="@+id/phone"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="8dp"  
 android:hint="Введите номер"  
 android:inputType="text|number"  
 app:layout\_constraintBottom\_toTopOf="@+id/button2"  
 tools:ignore="MissingConstraints"  
 tools:layout\_editor\_absoluteX="0dp" />  
  
 <Button  
 android:id="@+id/button2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:onClick="add"  
 android:text="@string/strinf"  
 tools:ignore="MissingConstraints"  
 tools:layout\_editor\_absoluteX="16dp"  
 tools:layout\_editor\_absoluteY="60dp" />  
  
 <ListView  
 android:id="@+id/alv"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/button2"  
 tools:ignore="MissingConstraints"  
 tools:layout\_editor\_absoluteX="0dp" />  
 </LinearLayout>  
  
</android.support.constraint.ConstraintLayout>*

Листинг 3 – activity\_adapter\_db

*<?xml version="1.0" encoding="utf-8"?>  
<TextView xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/items"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:padding="10dp"  
 android:textSize="16sp" >  
</TextView>*

*</android.support.constraint.ConstraintLayout>*

Листинг 4 – items

*package com.example.ik.trdmeinworld;  
  
import android.content.Intent;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
import android.widget.CheckBox;  
import android.widget.ToggleButton;  
import android.widget.RadioButton;  
import android.view.KeyEvent;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 final EditText userName = (EditText) findViewById(R.id.user\_name);  
 userName.setOnKeyListener(new View.OnKeyListener() {  
 @Override  
 public boolean onKey(View v, int keyCode, KeyEvent event) {  
 if ((event.getAction() == KeyEvent.ACTION\_DOWN)  
 && (keyCode == KeyEvent.KEYCODE\_ENTER)) {  
 Toast.makeText(getApplicationContext(),  
 userName.getText(), Toast.LENGTH\_SHORT).show();  
 return true;  
 }  
 return false;  
 }  
 });  
  
 }  
  
 public void onButtonClicked(View v) {  
 Toast.makeText(this, "Кнопка нажата", Toast.LENGTH\_SHORT).show();  
 Intent intent = new Intent(this, DBActivity.class);  
 startActivity(intent);  
 }  
  
 public void onButtonClicked1(View view) {  
 Intent intent = new Intent(this, AdapterDBActivity.class);  
 startActivity(intent);  
 }  
  
 public void onCheckboxClicked(View v) {  
 if (((CheckBox) v).isChecked()) {  
 Toast.makeText(this, "Отмечено", Toast.LENGTH\_SHORT).show();  
 } else {  
 Toast.makeText(this, "Не отмечено", Toast.LENGTH\_SHORT).show();  
 }  
 }  
  
 public void onToggleClicked(View v) {  
 if (((ToggleButton) v).isChecked()) {  
 Toast.makeText(this, "Включено", Toast.LENGTH\_SHORT).show();  
 } else {  
 Toast.makeText(this, "Выключено", Toast.LENGTH\_SHORT).show();  
 }  
 }  
  
 public void onRadioButtonClicked(View v) {  
 RadioButton rb = (RadioButton) v;  
 Toast.makeText(this, "Выбрано животное: " + rb.getText(),  
 Toast.LENGTH\_SHORT).show();  
 }  
  
 public void clean(View view) {  
 EditText userName = (EditText) findViewById(R.id.user\_name);  
 userName.setText("", TextView.BufferType.EDITABLE);  
 }  
  
  
}*

Листинг 5 – MainActivity

*package com.example.ik.trdmeinworld;  
  
import android.app.ActionBar;  
import android.app.ListActivity;  
import android.content.Context;  
import android.database.Cursor;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.ContextMenu;  
import android.view.MenuItem;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.ListView;  
import android.widget.SimpleCursorAdapter;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import java.lang.reflect.Array;  
import java.util.Arrays;  
import java.util.LinkedList;  
import java.util.List;  
  
public class DBActivity extends AppCompatActivity {  
 DatabaseHandler db;  
 CharSequence text;  
 private int CM\_DELETE\_ID = 1;  
 ListView list;  
 List<String> contacts;  
 ArrayAdapter<String> aa;  
 int pos;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_db);  
 db = new DatabaseHandler(this);  
 for (int i = 0; i <= 4; i++)  
 db.addContact(new Contact("8029599650" + i));  
 contacts = new LinkedList<>(Arrays.asList(db.getAllContacts()));  
 aa = new ArrayAdapter<String>(this, R.layout.items, contacts);  
 list = findViewById(R.id.lv);  
 list.setAdapter(aa);  
 registerForContextMenu(list);  
 list.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 public void onItemClick(AdapterView<?> parent, View v, int position, long id) {  
 text = db.getContact(String.valueOf(((TextView) v).getText()));  
 int duration = Toast.LENGTH\_LONG;  
 Context context = getApplicationContext();  
 Toast.makeText(context, text, duration).show();  
 }  
 });  
  
 }  
  
 public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {  
 super.onCreateContextMenu(menu, v, menuInfo);  
 menu.add(0, CM\_DELETE\_ID, 0, R.string.delete\_record);  
 AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;  
 pos = info.position;  
 }  
  
 @Override  
 public boolean onContextItemSelected(MenuItem item) {  
 if(item.getItemId() == CM\_DELETE\_ID)  
 {  
 AdapterView.AdapterContextMenuInfo acmi = (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();  
 db.deleteContact(contacts.get((int)acmi.id));  
 contacts.remove((int)acmi.id);  
 this.aa.notifyDataSetChanged();  
 return true;  
 }  
 return super.onContextItemSelected(item);  
 }  
  
 @Override  
 protected void onPause() {  
 super.onPause();  
 db.deleteAll();  
 }  
  
 @Override  
 protected void onDestroy() {  
 super.onDestroy();  
 db.deleteAll();  
 }  
}*

Листинг 6 – DBActivity

*package com.example.ik.trdmeinworld;  
  
import android.database.Cursor;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.ContextMenu;  
import android.view.MenuItem;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.EditText;  
import android.widget.ListView;  
import android.widget.SimpleCursorAdapter;  
  
public class AdapterDBActivity extends AppCompatActivity {  
 AdapterHandler db;  
 Cursor cursor;  
 SimpleCursorAdapter adapter;  
 private int CM\_DELETE\_ID = 1;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_adapter\_db);  
 db = new AdapterHandler(this);  
 db.open();  
 cursor = db.getAllContacts();  
 startManagingCursor(cursor);  
 String[] from = new String[] {db.phone};  
 int[] to = new int[] { R.id.items};  
 adapter = new SimpleCursorAdapter(this, R.layout.items, cursor, from, to);  
 ListView view = findViewById(R.id.alv);  
 view.setAdapter(adapter);  
 registerForContextMenu(view);  
 }  
  
 @Override  
 public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {  
 super.onCreateContextMenu(menu, v, menuInfo);  
 menu.add(0, CM\_DELETE\_ID, 0, R.string.delete\_record);  
 }  
  
 @Override  
 public boolean onContextItemSelected(MenuItem item) {  
 if(item.getItemId() == CM\_DELETE\_ID)  
 {  
 AdapterView.AdapterContextMenuInfo acmi = (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();  
 db.delContact(acmi.id);  
 cursor.requery();  
 return true;  
 }  
 return super.onContextItemSelected(item);  
 }  
  
 @Override  
 protected void onPause() {  
 super.onPause();  
 db.close();  
 }  
  
 public void add(View view) {  
 EditText text = findViewById(R.id.phone);  
 db.addContact(new Contact(text.getText().toString()));  
 cursor.requery();  
 }  
}*

Листинг 7 – AdapterDBActivity

*package com.example.ik.trdmeinworld;  
  
import android.content.ContentValues;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.content.Context;  
  
public class DatabaseHandler extends SQLiteOpenHelper{  
  
 private static final int version = 1;  
 private static final String name = "contactsManager";  
 private String contacts = "contacts";  
 public String id = "id";  
 public String phoneNumber = "phoneNumber";  
  
  
  
 public DatabaseHandler(Context context) {  
 super(context, name, null, version);  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 String create = "create table " + contacts + "(" + id + " integer primary key autoincrement, " +phoneNumber + " text" + ")";  
 db.execSQL(create);  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 db.execSQL("drop table if exists " + contacts);  
 onCreate(db);  
 }  
  
 public void addContact(Contact contact) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(phoneNumber, contact.getPhoneNumber());  
 db.insert(contacts, null, values);  
 db.close();  
 }  
  
 public String getContact(String phone) {  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.query(contacts, new String[] { id, phoneNumber }, phoneNumber + "=?",  
 new String[] { String.valueOf(phone) }, null, null, null, null);  
  
 if (cursor != null){  
 cursor.moveToFirst();  
 }  
 return cursor.getString(0);  
 }  
  
 public String[] getAllContacts() {  
 String selectQuery = "select \* from " + contacts;  
 String[] contact = new String[0];  
 SQLiteDatabase db = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, null);  
  
 if (cursor.moveToFirst()) {  
 do {  
 contact = increaseArray(contact);  
 contact[contact.length - 1] = cursor.getString(1);  
 } while (cursor.moveToNext());  
 }  
 return contact;  
 }  
  
 private String[] increaseArray(String[] theArray)  
 {  
 int i = theArray.length;  
 int n = ++i;  
 String[] newArray = new String[n];  
 for(int cnt=0;cnt<theArray.length;cnt++)  
 newArray[cnt] = theArray[cnt];  
 return newArray;  
 }  
  
 public void deleteContact(String number) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(contacts, phoneNumber + " = ?", new String[] { String.valueOf(number) });  
 db.close();  
 }  
  
 public void deleteAll() {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(contacts, null, null);  
 db.close();  
 }  
}*

Листинг 8 – DatabaseHandler

*package com.example.ik.trdmeinworld;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.os.Build;  
import android.support.annotation.RequiresApi;  
  
public class AdapterHandler {  
 private static final int version = 1;  
 private static final String name = "contactsManagers";;  
 private static final String table = "contacts";  
 public static final String id = "\_id";  
 public static final String phone = "phoneNumber";  
 private static final String create = "create table " + table + "(" + id + " integer primary key autoincrement, " + phone + " text" + ")";  
  
 private Context context;  
 private DBHandler handler;  
 private SQLiteDatabase db;  
  
 public AdapterHandler(Context context)  
 {  
 this.context = context;  
 }  
  
 public void open()  
 {  
 handler = new DBHandler(context, name, null, version);  
 db = handler.getWritableDatabase();  
 }  
  
 public void close()  
 {  
 if(db != null) handler.close();  
 }  
  
 public void addContact(Contact contact) {  
 ContentValues values = new ContentValues();  
 values.put(phone, contact.getPhoneNumber());  
 db.insert(table, null, values);  
 }  
  
 public Cursor getAllContacts()  
 {  
 return db.query(table, new String[] {id, phone}, null, null, null, null, null, null);  
 }  
  
 public void delContact(long \_id)  
 {  
 db.delete(table, id + " = " + \_id, null);  
 }  
  
 public void deleteAll() {  
 db.delete(table, null, null);  
 }  
  
 private class DBHandler extends SQLiteOpenHelper  
 {  
  
 public DBHandler(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {  
 super(context, name, factory, version);  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 db.execSQL(create);  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 db.execSQL("drop table if exists " + table);  
 onCreate(db);  
 }  
 }  
}*

Листинг 9 – AdapterHandler

*<resources>  
 <string name="app\_name">TrdMeinWorld</string>  
 <string name="strinf">Добавить</string>  
 <string name="delete\_record">Удалить</string>  
</resources>*

Листинг 10– Установка текста для приложения

*<?xml version="1.0" encoding="utf-8"?>  
<selector xmlns:android="http://schemas.android.com/apk/res/android">  
 <item android:drawable="@android:color/holo\_blue\_dark"  
 android:state\_pressed="true"/>  
 <item android:drawable="@android:color/holo\_green\_dark"  
 android:state\_focused="true"/>  
 <item android:drawable="@android:color/holo\_red\_dark" />  
</selector>*

Листинг 11– smile\_button

*<?xml version="1.0" encoding="utf-8"?>  
<selector xmlns:android="http://schemas.android.com/apk/res/android">  
 <item android:drawable="@android:color/holo\_red\_dark"  
 android:state\_pressed="true"/>  
 <item android:drawable="@android:color/holo\_green\_dark"  
 android:state\_focused="true"/>  
 <item android:drawable="@android:color/holo\_orange\_dark" />  
</selector>*

Листинг 12 – smile\_button1

**Резулътат проделанной работы:**

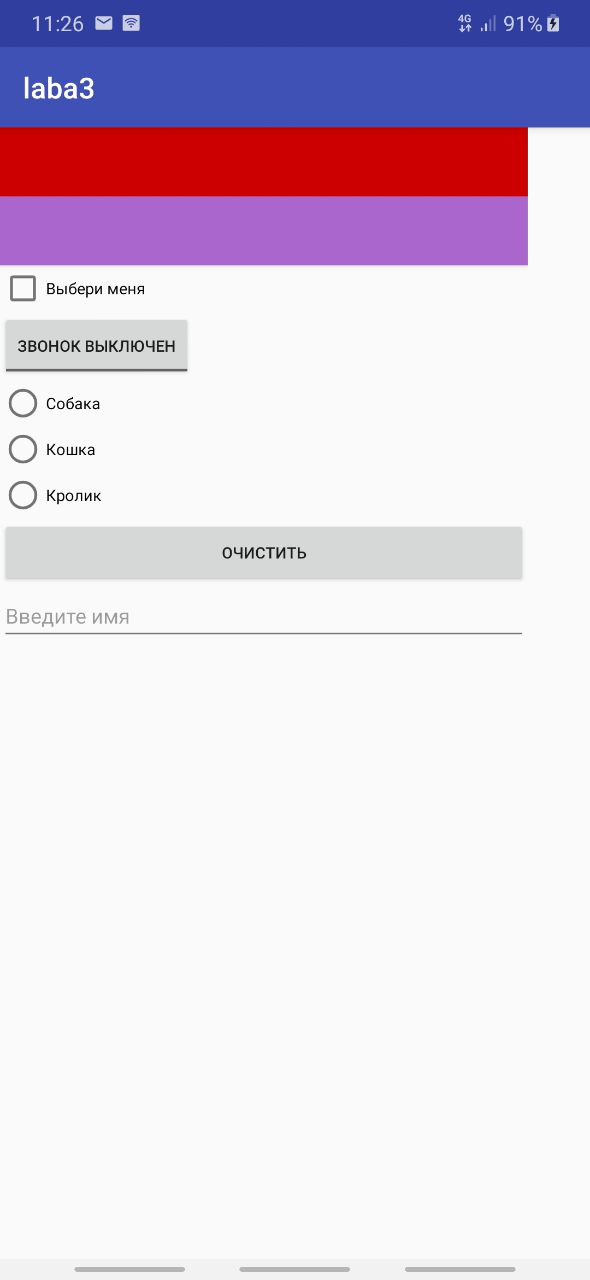


Рисунок 1 – Основное окно

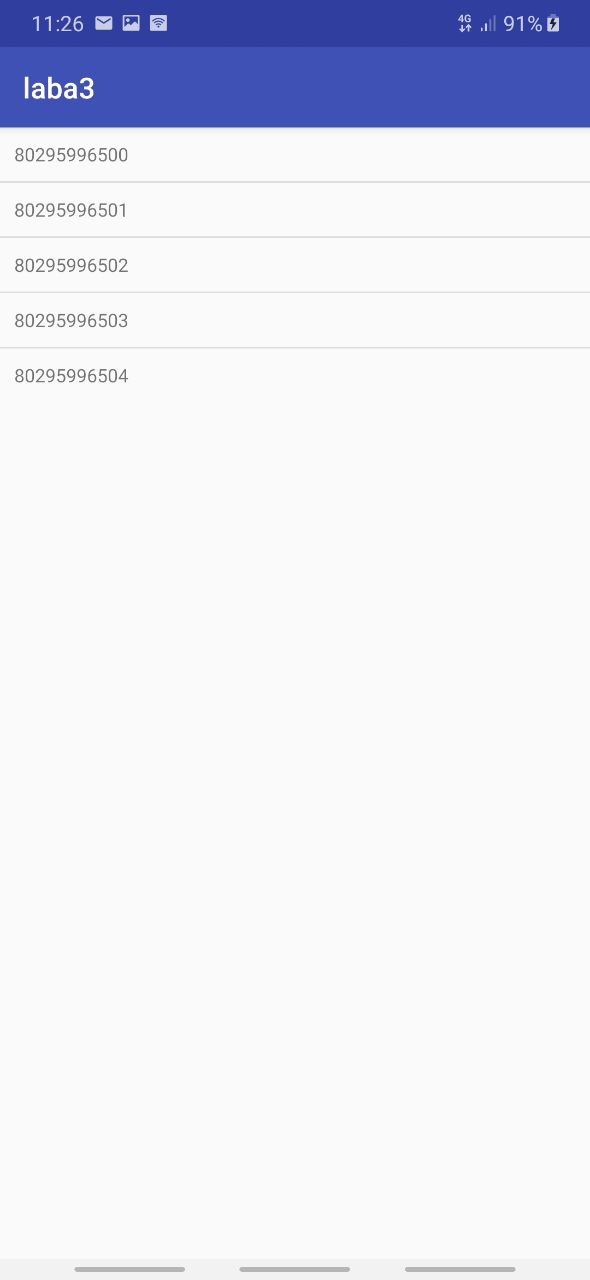


Рисунок 2 – DBActivity



Рисунок 2 – Удаление строки

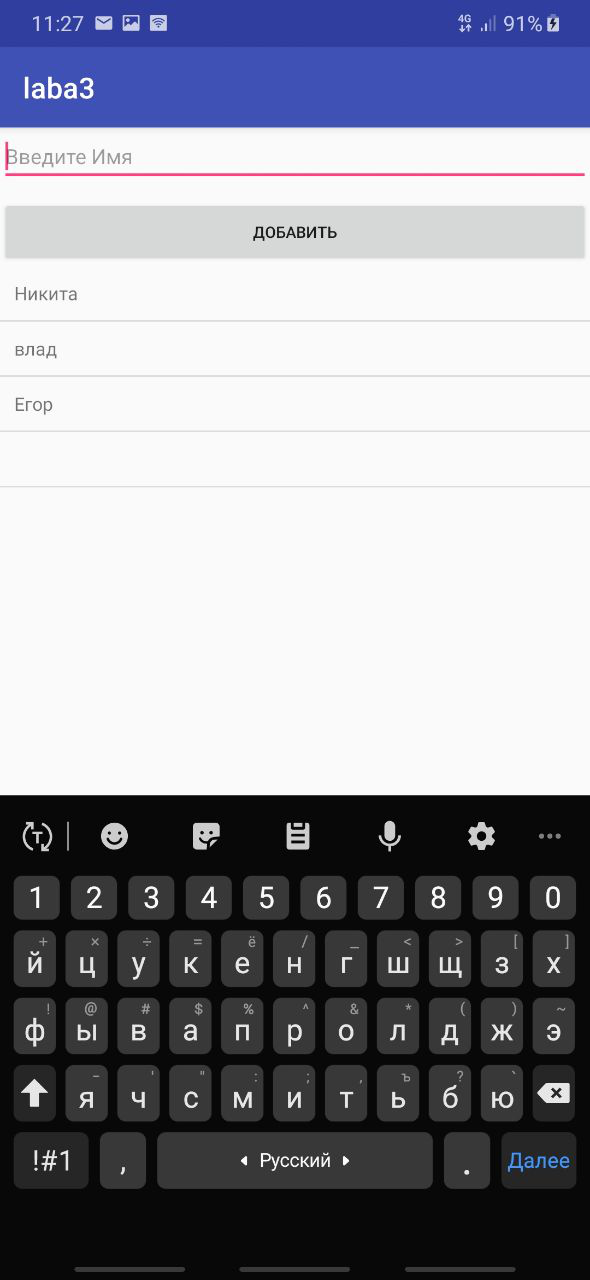


Рисунок 3 – AdapterDBActivity

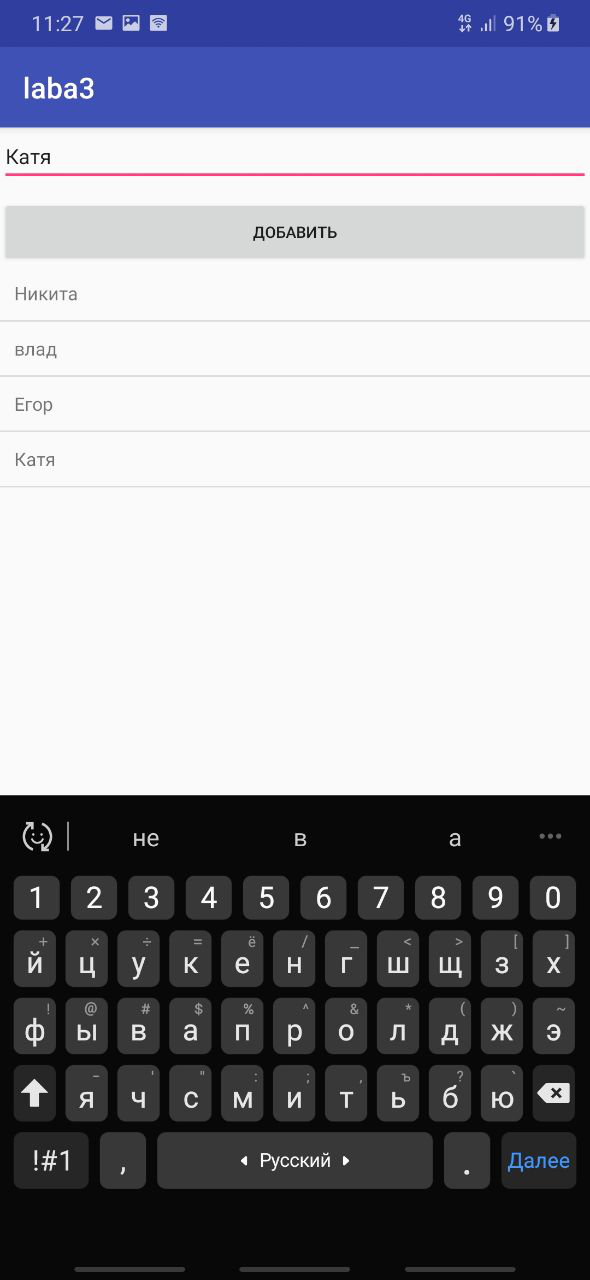


Рисунок 4 – Добавление строки

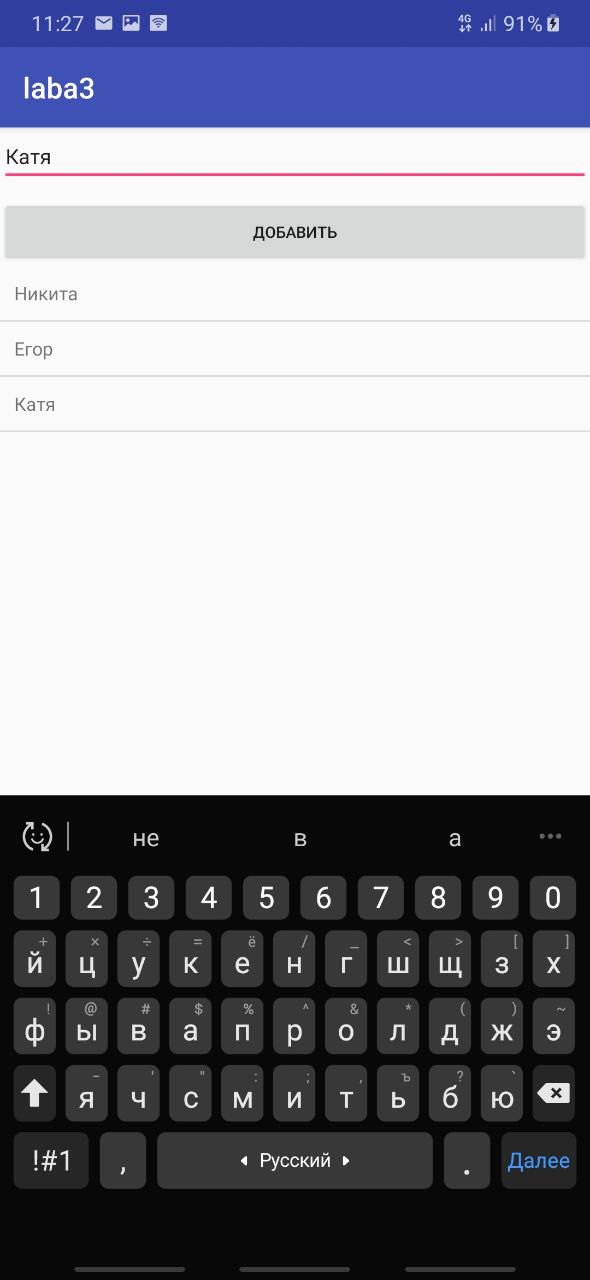


Рисунок 5 – Удаление строки

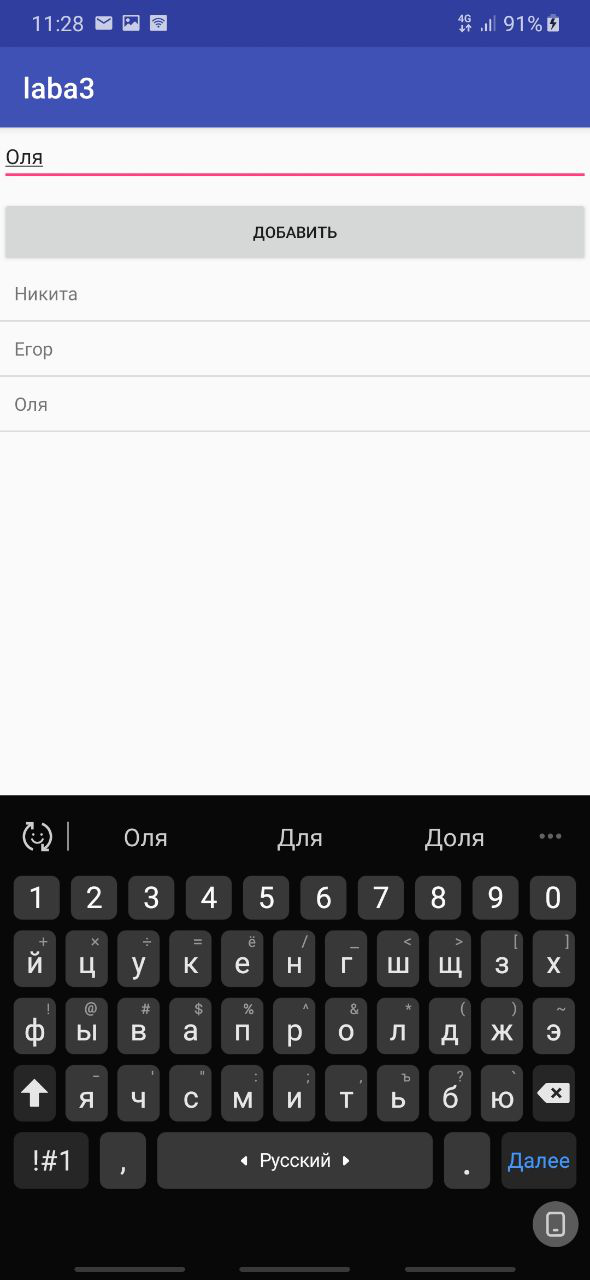
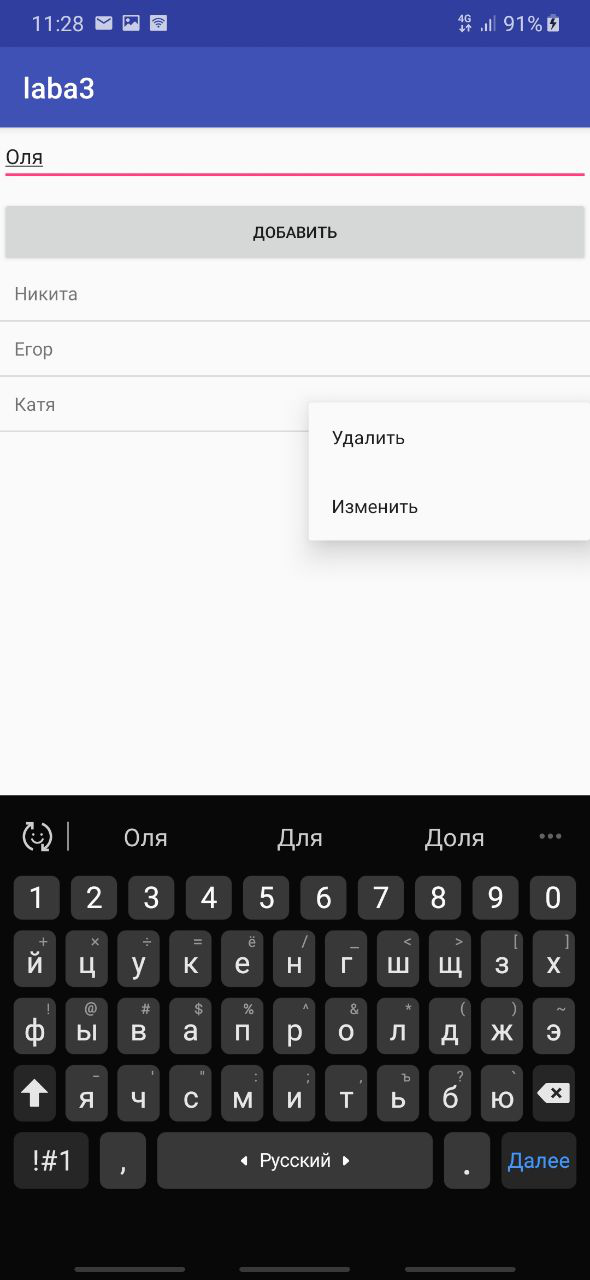


Рисунок 5 – Изменение строки

**Вывод**: В ходе лабораторной работы было разработано приложение, показывающее необходимые элементы и работающее с БД.