МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Лабораторная работа № 10

По дисциплине: «Разработка программного обеспечения для мобильных платформ»

Тема: «Фрагменты. ViewPager. Хранение информации в базе данных SQLite»

Выполнила: Велесевич В. А.

Группа: ПО-10

Проверил: Козинский А. А.

Брест 2025

**Цель:** научиться работать с базой данных и фрагментами.

**Постановка задачи:**

**Одним из доступных способов заранее подготовьте базу данных.** База данных, содержащая менее 20 записей будет считаться отсутсвующей.

1. Разработать приложение MyNotes, представляющее собой View Pager.

2. Поместить в View Pager четыре фрагмента: FragmentShow, FragmentAdd, FragmentDel, FragmentUpdate.

3. В View Pager добавить верхнее меню вкладок (PagerTabStrip) с заголовками Show, Add, Del, Update.

4. Во фрагменте FragmentShow реализовать кастомизированный список заметок ListView с помощью собственного адаптера.

5. В каждом пункте списка отобразить следующую информацию о заметке пользователя: номер, описание заметки.

6. Хранение, а также предоставление информации о заметках адаптеру реализовать с помощью базы данных SQLite.

7. Во фрагменте FragmentAdd реализовать функционал добавления новой заметки посредством ввода описания заметки в поле EditText и добавления информации в базу данных SQLite по нажатию кнопки Add.

8. Во фрагменте FragmentDel реализовать функционал удаления новой заметки посредством ввода ее номера в поле EditText и удаления информации из базы данных SQLite по нажатию кнопки Del.

9. Во фрагменте FragmentUpdate реализовать функционал **обновления** существующей заметки посредством ввода ее номера в поле EditText, ввода нового описания в поле EditText и обновления информации в базе данных SQLite по нажатию кнопки Update.

10. База данных, содержащая менее 20 записей будет считаться отсутсвующей.

**Графическое изображение интерфейса:**

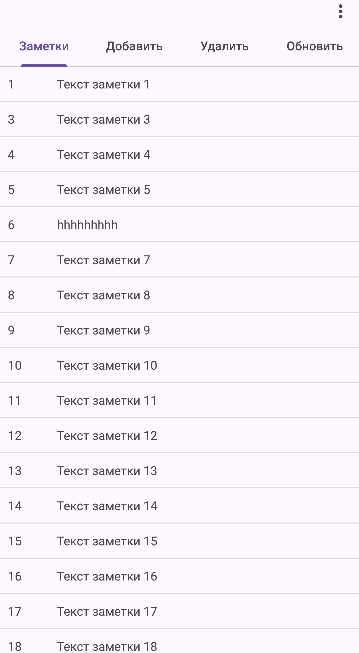


Рис. 1 - Кастомизированный список заметок ListView с помощью собственного адаптера

Рис. 2 - Отображение детальной информации об элементе на отдельном экране внутри отдельного фрагмента

**Код программы:**

**MainActivity**  
package com.example.lab33;  
import android.os.Bundle;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.lifecycle.ViewModelProvider;  
import androidx.viewpager2.widget.ViewPager2;  
import com.example.lab33.R.id;  
import com.example.lab33.R.layout;  
import com.google.android.material.tabs.TabLayout;  
import com.google.android.material.tabs.TabLayoutMediator;  
  
public class MainActivity extends AppCompatActivity {  
 private ViewPager2 viewPager;  
 private FragmentAdapter fragmentAdapter;  
 private SharedViewModel sharedViewModel;  
  
 public MainActivity() {  
 }  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 this.setContentView(layout.activity\_main);  
 this.sharedViewModel = (SharedViewModel)(new ViewModelProvider(this)).get(SharedViewModel.class);  
 this.viewPager = (ViewPager2)this.findViewById(id.viewPager);  
 this.fragmentAdapter = new FragmentAdapter(this);  
 this.viewPager.setAdapter(this.fragmentAdapter);  
 TabLayout tabLayout = (TabLayout)this.findViewById(id.tabLayout);  
 (new TabLayoutMediator(tabLayout, this.viewPager, (tab, position) -> {  
 switch (position) {  
 case 0:  
 tab.setText("Show");  
 break;  
 case 1:  
 tab.setText("Add");  
 break;  
 case 2:  
 tab.setText("Del");  
 break;  
 case 3:  
 tab.setText("Update");  
 }  
  
 })).attach();  
 }  
  
 public SharedViewModel getSharedViewModel() {  
 return this.sharedViewModel;  
 }  
}

**Fragmentadapter**

package com.example.lab33;  
  
import androidx.annotation.NonNull;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentActivity;  
import androidx.viewpager2.adapter.FragmentStateAdapter;  
  
public class FragmentAdapter extends FragmentStateAdapter {  
 public FragmentAdapter(@NonNull FragmentActivity fragmentActivity) {  
 super(fragmentActivity);  
 }  
  
 @NonNull  
 public Fragment createFragment(int position) {  
 switch (position) {  
 case 0:  
 return new FragmentShow();  
 case 1:  
 return new FragmentAdd();  
 case 2:  
 return new FragmentDel();  
 case 3:  
 return new FragmentUpdate();  
 default:  
 return new FragmentShow();  
 }  
 }  
  
 public int getItemCount() {  
 return 4;  
 }  
  
 public void refreshFragments() {  
 this.notifyDataSetChanged();  
 }  
}

**DatabaseHelper**  
package com.example.lab33;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.util.Log;  
import java.util.ArrayList;  
import java.util.List;  
  
public class DatabaseHelper extends SQLiteOpenHelper {  
 private static final String DATABASE\_NAME = "notes.db";  
 private static final int DATABASE\_VERSION = 1;  
 private static final String TABLE\_NAME = "notes";  
 private static final String COLUMN\_ID = "id";  
 private static final String COLUMN\_DESCRIPTION = "description";  
  
 public DatabaseHelper(Context context) {  
 super(context, "notes.db", (SQLiteDatabase.CursorFactory)null, 1);  
 }  
  
 public void onCreate(SQLiteDatabase db) {  
 String query = "CREATE TABLE notes (id INTEGER PRIMARY KEY AUTOINCREMENT, description TEXT);";  
 db.execSQL(query);  
 Log.d("DatabaseHelper", "База данных и таблица созданы");  
  
 for(int i = 1; i <= 30; ++i) {  
 ContentValues values = new ContentValues();  
 values.put("description", "Готовая заметка " + i);  
 long result = db.insert("notes", (String)null, values);  
 if (result == -1L) {  
 Log.d("DatabaseHelper", "Ошибка при добавлении готовой заметки " + i);  
 } else {  
 Log.d("DatabaseHelper", "Готовая заметка добавлена: " + i);  
 }  
 }  
  
 }  
  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 db.execSQL("DROP TABLE IF EXISTS notes");  
 this.onCreate(db);  
 }  
 public void addNote(String description) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put("description", description);  
 long result = db.insert("notes", (String)null, values);  
 if (result == -1L) {  
 Log.d("DatabaseHelper", "Ошибка при добавлении заметки");  
 } else {  
 Log.d("DatabaseHelper", "Заметка добавлена: " + description);  
 }  
  
 }  
 public List<Note> getAllNotes() {  
 List<Note> notes = new ArrayList();  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.rawQuery("SELECT \* FROM notes", (String[])null);  
 if (cursor.moveToFirst()) {  
 do {  
 Note note = new Note();  
 note.setId(cursor.getInt(0));  
 note.setDescription(cursor.getString(1));  
 notes.add(note);  
 } while(cursor.moveToNext());  
 }  
 cursor.close();  
 return notes;  
 }  
 public void deleteNote(int id) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete("notes", "id = ?", new String[]{String.valueOf(id)});  
 }  
 public void updateNote(int id, String description) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put("description", description);  
 db.update("notes", values, "id = ?", new String[]{String.valueOf(id)});  
 }  
}

**Вывод:** реализовала интерфейс приложения для отображения списка элементов.