МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Звіт з лабораторної роботи №2

«Список нотаток з використанням бази даних SQLite»

з дисципліни «Програмування під Android»

Виконала:

ст. групи ПI-15-1

Михневич Т. К.

Прийняв:

ст. викл. каф. ПІ

Сокорчук І. П.

Харків 2018

**Завдання**

Для кожної нотатки задати назву, опис, важливість (три класи), час и дата призначення і картинка (вибирається з галереї). У списку нотатки відображаються з іконкою галереї і часом створення (праворуч зверху більш дрібним шрифтом), а також графічним відображенням класу важливості (іконка з множини). Реалізувати функції додавання, видалення, редагування (зі збереженням між перезапусками додатку) нотаток. Видалення і редагування викликаються з контекстного меню при довгому натисненні, додавання – з основного меню ActionBar. У ActionBar реалізувати пошук / фільтрацію заміток за змістом тексту, а також фільтрацію за важливістю. Текстові елементи користувацького інтерфейсу локалізовані для української та англійської мов.

**Опис додатку**

У результаті виконання лабораторної заняття був розроблений додаток для роботи з нотатками.

Після запуску додатку відкривається головний екран (див. рис. 1). Після додавання нотаток вони будуть відображені на ньому у вигляді списку.

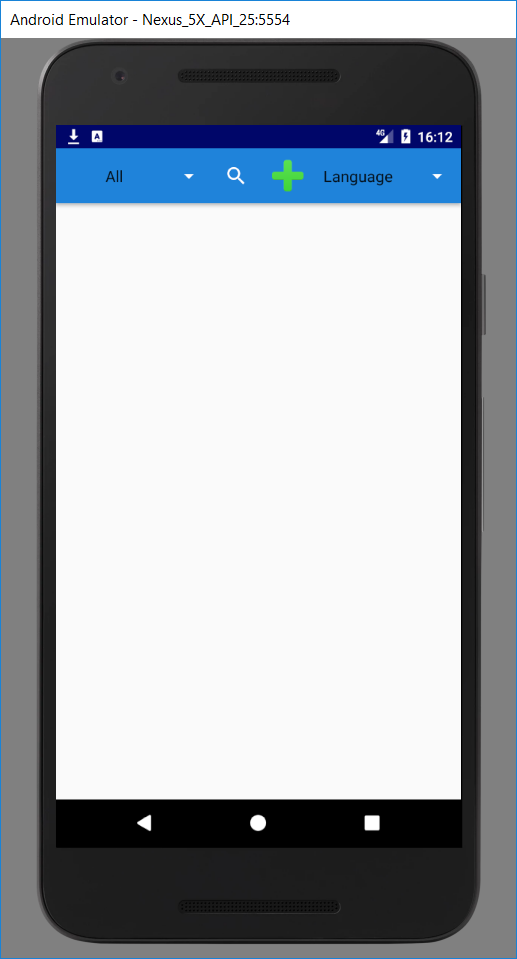


Рисунок 1 – Головний екран додатку одразу після запуску додатку

На головному екрані присутня кнопка «+» для додавання нової нотатки. Після натиснення на цю кнопку відкривається форма додавання нотатки (див. рис. 2).

Форма також використовується і для редагування існуючих нотаток.

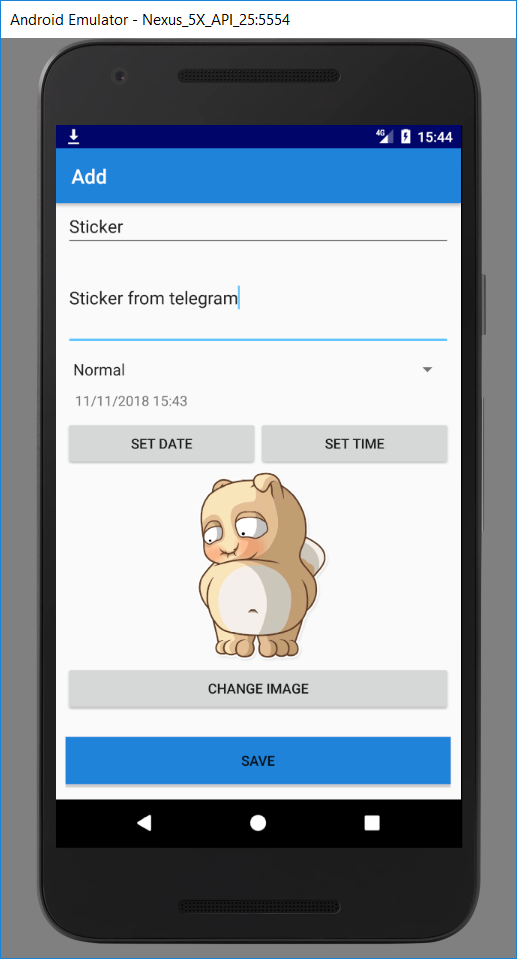


Рисунок 2 – Форма додавання нової нотатки

На формі можна обрати важливість, зображення, дату та час призначення, назву та опис.

Після натиснення на кнопку для зміни дати або часу можна обрати потрібне значення (див рис.3, 4).

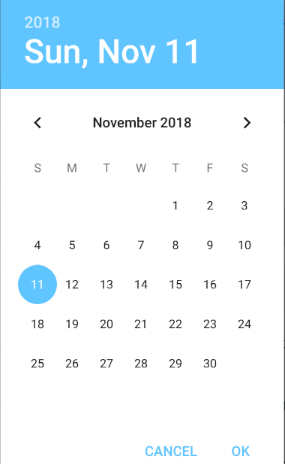
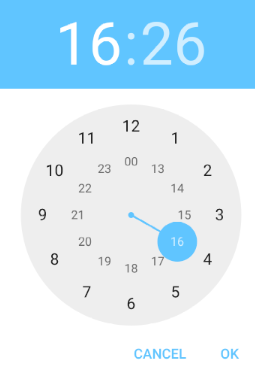
 

Рисунок 3 – Вибір дати Рисунок 4 – Вибір часу

Також існує кнопка для зміни зображення, пісня натиснення на яку можна вибрати зображення з галереї мобільного пристрою.

Після додавання нотаток на головному екрані додатка можна побачити список (див рис. 6).

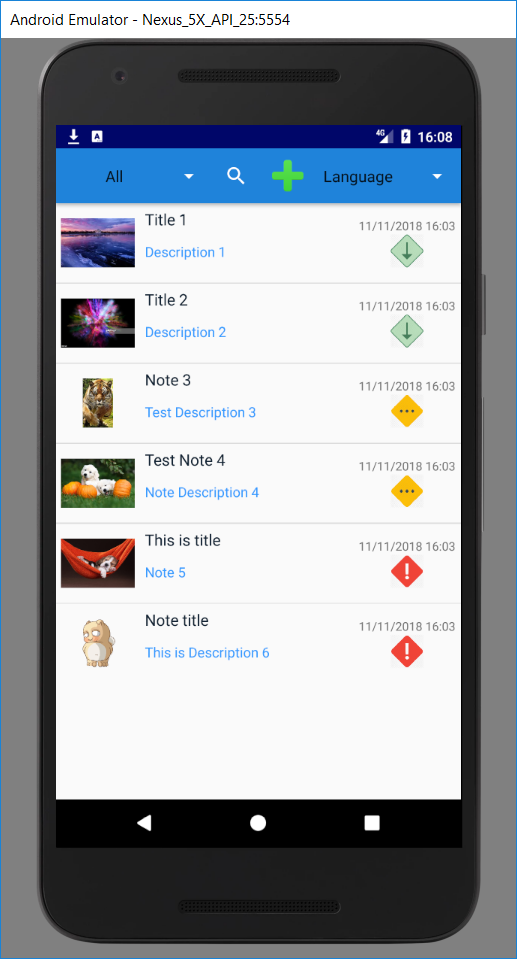


Рисунок 6 – Головний екран додатку зі списком нотаток

В додатку також реалізована функція фільтрації нотаток за важливість та пошук за змістом. Для цього треба вибрати відповідний елемент зі списку із головного меню та / або написати ключові слова для пошуку (див. рис. 7).

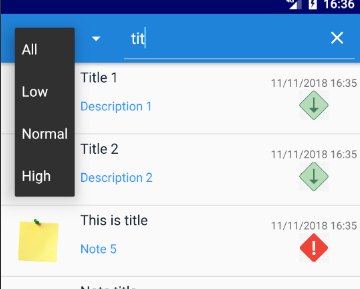


Рисунок 7 – Фільтрація та пошук нотаток

Для редагування або видалення нотатки, треба натиснути та утримати палець на відповідній нотатці (див. рис. 8). В результаті користувач побачить дві можливості – редагування та видалення.

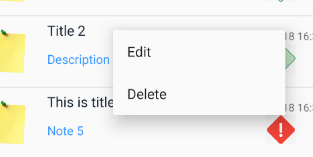


Рисунок 8 – Редагування та видалення

Для зміни мови додатку на українську потрібно обрати мову у головному меню. Після цього інтерфейс додатку буде відображений на українській мові (див. рис. 9).

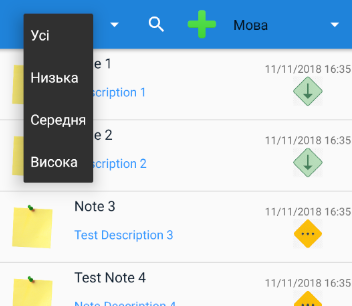


Рисунок 9 - Локалізація

**Опис реалізації**

Для збереження нотаток в мобільному додатку було використано SQLite базу даних.

База даних складається з однієї таблиці, тож розглянемо її структуру (див. рис. 10).

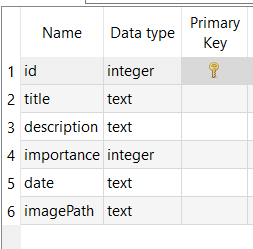


Рисунок 10 – Структура таблиці Notes

Для роботи з цією базою даних був розроблений NotesDbHelper, що розширює SQLiteOpenHelper (див. фрагмент коду 1). В цьому класі описується логіка для створення таблиці, що буде зберігати дані. Тобто, при першому звернені до бази даних буде викликаний метод створення бази даних, тому що її ще не існує. Метод оновлення бази даних не використовується в даному випадку, тому що це перша версія бази даних.

**package** ua.nure.notesapp.helpers;  
  
**import** android.content.Context;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
  
**public class** NotesDbHelper **extends** SQLiteOpenHelper {  
  
 **public** NotesDbHelper(Context context) {  
 **super**(context, **"notesDb"**, **null**, 1);  
 }  
  
 @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 db.execSQL(**"create table notes ("** + **"id integer primary key autoincrement,"** + **"title text,"** + **"description text,"** + **"importance integer,"** + **"date text,"** + **"imagePath text"** + **");"**);  
 }  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion)

{  
  
 }  
}

Фрагмент коду 1 – клас NotesDbHelper

Для безпосереднього доступу до даних був створений клас NotesStore що містить в собі всю логіку, необхідну для взаємодії з базою даних (див. фрагмент коду 2). В даному класі є методи для отримання, додавання, оновлення та видалення даних, а також метод фільтрації та пошуку.

Для роботи з базою використовується курсор, що перебирає рядки результатів. Після цього повертає результат. Для запитів, що потребують ідентифікатора (оновлення та видалення), передається параметр «selection», що містить в собі критерії пошуку.

Також, фільтрація та пошук виконуються на рівні бази даних. Для цього динамічно формуються критерії пошуку, що будуть передані до запиту.

**public class** NotesStore {  
 **private** String **dateFormat** = **"MM/dd/yyyy HH:mm"**;  
 **private** NotesDbHelper **\_dbHelper**;  
 **private** SQLiteDatabase **\_db**;  
  
 **public** NotesStore(NotesDbHelper dbHelper) {  
 **this**.**\_dbHelper** = dbHelper;  
 **\_db** = dbHelper.getWritableDatabase();  
 }  
  
 **public** ArrayList<Note> getAll() {  
 Cursor c = **\_db**.query(**"notes"**, **null**, **null**, **null**, **null**, **null**, **null**);  
 ArrayList<Note> result = **new** ArrayList<>();  
  
 **if** (c.moveToFirst()) {  
 **do** {  
 Note note = parseNote(c);  
 result.add(note);  
 } **while** (c.moveToNext());  
 }  
 c.close();  
  
 **return** result;  
 }  
  
 **public** ArrayList<Note> searchAndFilter(**int** importance, String searchText)

{  
 **boolean** shouldFilter = importance < 4 && importance > 0;  
 **boolean** shouldSearch = searchText.length() > 0;  
  
 String importanceStringValue = String.*valueOf*(importance);  
  
 String query = **""**;  
 **if** (shouldFilter) query += **"importance = "** + importanceStringValue;  
  
 **if** (shouldSearch) {  
 **if** (!query.equals(**""**)) {  
 query = **"("** + query + **") and"**;  
 }  
 query += **"(title like '%"** + searchText + **"%' or description like '%"** + searchText + **"%')"**;  
 }  
  
 Cursor c = **\_db**.query(**"notes"**, **null**, query, **null**, **null**, **null**, **null**);  
 ArrayList<Note> result = **new** ArrayList<>();  
  
 **if** (c.moveToFirst()) {  
 **do** {  
 Note note = parseNote(c);  
 result.add(note);  
 } **while** (c.moveToNext());  
 }  
 c.close();  
 **return** result;  
 }  
  
 **public** Note getById(**int** id) {  
 Cursor c = **\_db**.query(**"notes"**, **null**, **"id = ?"**, **new** String[]{String.*valueOf*(id)}, **null**, **null**, **null**);  
 Note note = **null**;  
  
 **if** (c.moveToFirst()) {  
 note = parseNote(c);  
 }  
 c.close();  
 **return** note;  
 }  
  
 **public void** update(Note note) {  
 ContentValues cv = getContentValuesByNote(note);  
 **int** updCount = **\_db**.update(**"notes"**, cv, **"id = ?"**,  
 **new** String[]{String.*valueOf*(note.getId())});  
 }  
  
 **public void** add(Note note) {  
 ContentValues cv = getContentValuesByNote(note);  
 **long** rowID = **\_db**.insert(**"notes"**, **null**, cv);  
 note.setId((**int**) rowID);  
 }  
  
 **public void** delete(**int** id) {  
 **int** delCount = **\_db**.delete(**"notes"**, **"id = ?"**, **new** String[]{String.*valueOf*(id)});

}  
}

Фрагмент коду 2 – клас NotesStore

**Висновки**

У результаті виконання лабораторної роботи був розроблений додаток для роботи з нотатками з функціями додавання, редагування, видалення (зі збереженням нотаток між перезапусками додатку), пошуку та фільтрації.

Для реалізації були використані 2 створені Activity, а також такі елементи, як ListView, Adapter, SearchView, DatePickerDialog, TimePickerDialog. У якості бази даних була використана SQLite.