МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Звіт

з лабораторної роботи № 5 з дисципліни

Програмування під Android

Виконала: Перевірив:

ст. гр. ПЗПІ-23-5 ст. викладач кафедри ПІ

Євсєєва Олександра Сокорчук Ігор Петрович

Харків 2024

ЗАВДАННЯ

Доповнити завдання №4 наступними функціями:

реалізувати різне відображення для портретної та ландшафтної орієнтації екрану за допомогою тих самих фрагментів;

створити дві редаговані теми (тло, гарнітура шрифту, колір тексту тощо) і застосовувати їх в залежності від налаштувань програми.

додати можливість задавати в налаштуваннях програми розмір шрифту.

Переробити завдання №4 так, щоб дані нотаток зберігалися у базі даних SQLite

ХІД РОБОТИ

Продовжуємо роботу над лабораторною роботою №4. Почнемо з першого пункту: додавання різного відображення для портретної та ландшафтної орієнтації екрану за допомогою тих самих фрагментів.

Для цього відкриваємо файли розмітки та створюємо нові фрагменти: для відображення списку, деталей та редагування нотаток. Від звичайного, вертикального, вони відрізняються горизонтальним елементами на екрані. Наприклад: у деталях дата відображається в іншій колонці справа, а не списком, як у вертикальній. Повністю код фрагментів можна подивитися у [ДОДАТОК А].

Перейдемо до другого пункту роботи: редаговані теми. Для цього створимо іншу тему, нехай, темну. У стандартній змінено колір та стиль тексту:

1.<item name="android:textColor">@color/blue</item>  
2.<item name="android:textStyle">italic</item>

У другій, що тільки що створили, ті ж елементи зробимо іншого кольору та стилю:

3.<item name="android:textColor">@color/white</item>  
4.<item name="android:textStyle">bold</item>

Колір можна встановити білим, оскільки сама тема унаслідується від темної теми, отже будь-який світлий буде виглядати нормально.

Тепер третій пункт, що треба було додати: можливість задавати в налаштуваннях програми розмір шрифту. Переходимо до попередньо створеного класу NoteDataSource. Додаємо 2 методи для збереження та зчитування розміру тексту: saveTextSize та getTextSize.

1.public void saveTextSize(Integer size) {  
2. SharedPreferences.Editor editor = sharedPreferences.edit();  
3. editor.putInt("size", size);  
4. editor.apply();}

5. public Integer getTextSize() {  
6. return sharedPreferences.getInt("size", 1);}

Потім в залежності від id розміру, встановлюємо шрифт у NoteDetailsFragment та інших:

7.int textSize = NoteDataSource.*getInstance*().getTextSize();  
8.if (textSize == 0) {  
9. binding.titleLabel.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 8);  
10. binding.title.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 10);  
11. binding.descLabel.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 8);  
12. binding.desc.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 10);  
13. binding.dateLabel.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 8);  
14. binding.date.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 10);

Тепер перейдемо до най об’ємної частини: переробити завдання №4 так, щоб дані нотаток зберігалися у базі даних SQLite.

Для цього скористаємося DbHelper, що розширює SQLiteOpenHelper. Першим кроком створимо саму базу:

15.private static final String *SQL\_CREATE\_ENTRIES* =  
16. "CREATE TABLE IF NOT EXISTS note (id INTEGER PRIMARY KEY, title TEXT, desc TEXT, type INTEGER, date INTEGER, icon INTEGER)";

Одразу створюємо:

17.public DbHelper(@Nullable Context context) {  
18. super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  
19. this.getWritableDatabase().execSQL(*SQL\_CREATE\_ENTRIES*);}

Як вже видно по базі, у об’єкта Note було створено параметр id. Тепер його можна знайти за унікальним номером, а не за назвою, як було у попередній роботі, лабораторній №4.

20.public class Note {  
21. private Integer id;  
22. private String title;  
23. private String desc;  
24. private Type type;  
25. private Date date;  
26. @DrawableRes private Integer icon;

Повертаємось до NoteDataSource тепер для роботи із базою даних. У конструкторі створюється база даних (DbHelper) та SharedPreferences для збереження даних користувача.

27.public static NoteDataSource getInstance(Context context) {  
28. if (*INSTANCE* == null) {  
29. *INSTANCE* = new NoteDataSource(context);  
30. }  
31. return *INSTANCE*;}

Метод створює екземпляр, якщо його ще не існує.

Метод getNotes використовує Cursor для отримання всіх записів з таблиці notes.

32.notes.add( new Note( cursor.getInt(cursor.getColumnIndex(DbHelper.ID)), 33.cursor.getString(cursor.getColumnIndex(DbHelper.TITLE)), 34.cursor.getString(cursor.getColumnIndex(DbHelper.DESC)), 35.Note.Type.values()[cursor.getType(cursor.getColumnIndex(DbHelper.TYPE))], 36.new Date(cursor.getLong(cursor.getColumnIndex(DbHelper.DATE))), 37.cursor.getInt(cursor.getColumnIndex(DbHelper.ICON)) )

getNoteById фільтрує записи за ID (WHERE ID = ?):

38.String selection = DbHelper.ID + "= ?";

Повертає перший знайдений результат:

39.notes.add(  
40. new Note(  
41. cursor.getInt(cursor.getColumnIndex(DbHelper.*ID*)),  
42. cursor.getString(cursor.getColumnIndex(DbHelper.*TITLE*)),  
43. cursor.getString(cursor.getColumnIndex(DbHelper.*DESC*)),  
44.Note.Type.*values*()[cursor.getType(cursor.getColumnIndex(DbHelper.*TYPE*))],  
45. new Date(cursor.getLong(cursor.getColumnIndex(DbHelper.*DATE*))),  
46. cursor.getInt(cursor.getColumnIndex(DbHelper.*ICON*))));

Метод save зберігає та оновлює нотатку. Він перевіряє, чи існує така нотатка та робить відповідно до знайденого id:

47.if (note.getId() == null) {  
48. db.insert(DbHelper.*TABLE\_NOTES*, null, values);  
49.} else {  
50. String selection = DbHelper.*ID* + "= ?";  
51. String[] selectionArgs = {note.getId().toString()};  
52. db.update(  
53. DbHelper.*TABLE\_NOTES*,  
54. values,  
55. selection,  
56. selectionArgs);

Останній необхідний метод для CRUD це видалення. Отже, час для функції deleteNote:

57.public void deleteNote(Integer id) {  
58. String selection = DbHelper.*ID* + "= ?";  
59. String[] selectionArgs = {id.toString()};  
60. db.delete(DbHelper.*TABLE\_NOTES*, selection, selectionArgs);}

Таким чином, всі завдання було виконано.

ВИСНОВКИ

Під час виконання лабораторної роботи я набула навичок використання бази даних SQLite у повноцінному проєкті з Android. Я навчилася використовувати фрагменти для різної розмітки та закріпила знання з тем для зміни кольорів, текстів тощо у всьому застосунку.

Відео-звіт на YouTube: <https://youtu.be/gkWul8qP8TA>.

ВИКОРИСТАНІ ДЖЕРЕЛА

1. Документація Andoid developers Save data using SQLite. URL: <https://developer.android.com/training/data-storage/sqlite> (дата звернення: 06.12.2024).
2. Документація Andoid developers SQLiteDatabase. URL: <https://developer.android.com/reference/android/database/sqlite/SQLiteDatabase> (дата звернення: 06.12.2024).
3. Документація Andoid developers Save simple data with SharedPreferences URL: <https://developer.android.com/training/data-storage/shared-preferences> (дата звернення: 06.12.2024).
4. Документація Andoid developers Fragments URL: <https://developer.android.com/guide/fragments> (дата звернення: 06.12.2024).

ДОДАТОК А

Код програми

Увесь код програми можно знайти за посиланням у GitHub:

<https://github.com/NureIevsieievaOleksandra/android-pzpi-23-5-ievsieieva-oleksandra>.