**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до лабораторної роботи №7**

**з курсу**

**«Архітектура мобільних операційних систем і пристроїв»**

*Студентки 4 курсу*

*групи ПП-42*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

Матвіїв А.Ю.

*Викладач:*

Краснощок В.М.

**Київ – 2024**

***Тема:*** Робота з базами даних в ОС Android.

***Мета:*** Опанувати принципи роботи з СУБД SQLite без застосування спеціальних класів-адаптерів.

***Хід роботи:***

Створимо новий проект SQLTest в якому не будемо застосовувати спеціальні класи-адаптери.

MainActivity.kt

package com.example.sqltest

import android.content.ContentValues

import android.content.Context

import android.content.SharedPreferences

import android.database.sqlite.SQLiteDatabase

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.Toast

import androidx.activity.ComponentActivity

class MainActivity : ComponentActivity() {

   private lateinit var database: SQLiteDatabase

   private lateinit var listView: ListView

   private lateinit var adapter: ArrayAdapter<String>

   private val list = *mutableListOf*<String>()

   override fun onCreate(savedInstanceState: Bundle?) {

       super.onCreate(savedInstanceState)

       setContentView(R.layout.*activity\_main*)

       database = createDatabase(this)

       if (!areSampleDataInserted()) {

           insertSampleData("Tablet")

           insertSampleData("Smartphone")

           insertSampleData("Desktop Computer")

           insertSampleData("Laptop")

           setSampleDataInsertedFlag()

       }

       queryDatabase()

       listView = findViewById(R.id.*listView*)

       adapter = ArrayAdapter(this, android.R.layout.*simple\_list\_item\_1*, list)

       listView.*adapter* = adapter

       listView.setOnItemClickListener **{** \_, \_, position, \_ **->**

Toast.makeText(this, "Індекс $position: ${list[position]}", Toast.*LENGTH\_SHORT*).show()

**}**

registerForContextMenu(listView)

   }

   override fun onCreateContextMenu(

       menu: ContextMenu?, v: View?, menuInfo: ContextMenu.ContextMenuInfo?

   ) {

       super.onCreateContextMenu(menu, v, menuInfo)

*menuInflater*.inflate(R.menu.*context\_menu*, menu)  // Ваше контекстне меню

   }

   override fun onContextItemSelected(item: MenuItem): Boolean {

       val info = item.*menuInfo* as AdapterView.AdapterContextMenuInfo

       val position = info.position

       val itemName = list[position]

       return when (item.*itemId*) {

           R.id.*delete* -> {

               deleteUser(itemName)

               list.removeAt(position)

               adapter.notifyDataSetChanged()  // Оновлення адаптера для зміни списку

               Toast.makeText(this, "Елемент '$itemName' видалено", Toast.*LENGTH\_SHORT*).show()

               true

           }

           else -> super.onContextItemSelected(item)

       }

   }

   private fun createDatabase(context: Context): SQLiteDatabase {

       val dbName = "MyDatabase.db"

       return context.openOrCreateDatabase(dbName, Context.*MODE\_PRIVATE*, null).*apply* **{**

execSQL("CREATE TABLE IF NOT EXISTS Test (id INTEGER PRIMARY KEY AUTOINCREMENT, item TEXT)")

**}**

}

   private fun insertSampleData(item: String) {

       val values = ContentValues().*apply* **{**

put("item", item)

**}**

database.insert("Test", null, values)

   }

   private fun queryDatabase() {

       val cursor = database.rawQuery("SELECT \* FROM Test", null)

       if (cursor.moveToFirst()) {

           do {

               val itemIndex = cursor.getColumnIndex("item")

               if (itemIndex != -1) {

                   val item = cursor.getString(itemIndex)

                   list.add(item)

               }

           } while (cursor.moveToNext())

       }

       cursor.close()

   }

   private fun deleteUser(item: String) {

       database.delete("Test", "item = ?", *arrayOf*(item))

   }

   private fun areSampleDataInserted(): Boolean {

       val preferences: SharedPreferences = getSharedPreferences("AppPreferences", Context.*MODE\_PRIVATE*)

       return preferences.getBoolean("SampleDataInserted", false)

   }

   private fun setSampleDataInsertedFlag() {

       val preferences: SharedPreferences = getSharedPreferences("AppPreferences", Context.*MODE\_PRIVATE*)

       val editor = preferences.edit()

       editor.putBoolean("SampleDataInserted", true)

       editor.apply()

   }

   override fun onDestroy() {

       super.onDestroy()

       database.close()

   }

}

activity\_main.xml

<?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">

   <ListView

       android:id="@+id/listView"

       android:layout\_width="match\_parent"

       android:layout\_height="match\_parent" />

</LinearLayout>

context\_menu.xml

<!-- res/menu/context\_menu.xml -->

<menu xmlns:android="http://schemas.android.com/apk/res/android">

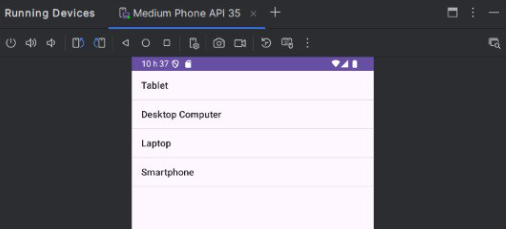
   <item

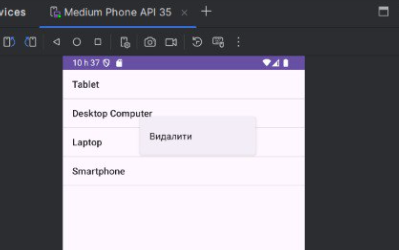
       android:id="@+id/delete"

       android:title="Видалити" />

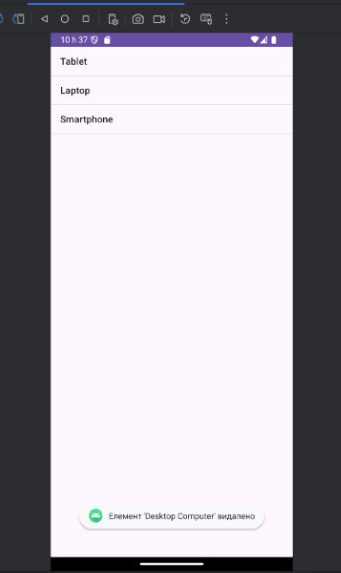
</menu>

Переглянемо результат роботи:





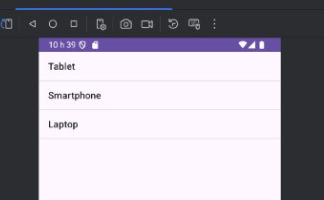
Бачимо повідомлення про видалення об’єкту:



Повністю вимкнем програму:



Бачимо, що всі дані збереглись:



Створимо новий проект і назвемо його «NotesSample»:

MainActivity.kt

package com.example.notessample

import android.os.Bundle

import android.view.ContextMenu

import android.view.MenuItem

import android.view.View

import android.widget.\*

import androidx.appcompat.app.AppCompatActivity

import com.example.notessample.R

class MainActivity : AppCompatActivity() {

   private lateinit var listView: ListView

   private lateinit var dbHandler: DataBaseHandler

   private lateinit var notesList: MutableList<Note>

   override fun onCreate(savedInstanceState: Bundle?) {

       super.onCreate(savedInstanceState)

       setContentView(R.layout.*activity\_main*)

       val add = findViewById<Button>(R.id.*add*)

       val note = findViewById<EditText>(R.id.*text*)

       listView = findViewById(R.id.*listView*)

       registerForContextMenu(listView)

       dbHandler = DataBaseHandler(this)

       loadNotes()

       add.setOnClickListener **{**

if (note.*text*.toString().*isNotEmpty*()) {

               val newNote = Note(text = note.*text*.toString())

               dbHandler.insertData(newNote)

               note.*text*.clear()

               loadNotes()

           } else {

               Toast.makeText(this, "Введіть текст", Toast.*LENGTH\_SHORT*).show()

           }

**}**

}

   private fun loadNotes() {

       notesList = dbHandler.readData()

       val adapter = ArrayAdapter(this, android.R.layout.*simple\_list\_item\_1*, notesList.*map* **{ it**.text **}**)

       listView.*adapter* = adapter

   }

   override fun onCreateContextMenu(menu: ContextMenu?, v: View?, menuInfo: ContextMenu.ContextMenuInfo?) {

       super.onCreateContextMenu(menu, v, menuInfo)

*menuInflater*.inflate(R.menu.*context\_menu*, menu)

   }

   override fun onContextItemSelected(item: MenuItem): Boolean {

       val info = item.*menuInfo* as AdapterView.AdapterContextMenuInfo

       val selectedNote = notesList[info.position]

       return when (item.*itemId*) {

           R.id.*delete* -> {

               dbHandler.deleteData(selectedNote.id)

               loadNotes()

               Toast.makeText(this, "Запис видалено", Toast.*LENGTH\_SHORT*).show()

               true

           }

           else -> super.onContextItemSelected(item)

       }

   }

}

activity\_main.xml

<?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"

   android:padding="16dp">

   <EditText

       android:id="@+id/text"

       android:layout\_width="match\_parent"

       android:layout\_height="wrap\_content"

       android:hint="Введіть новий запис"

       android:layout\_marginBottom="8dp"/>

   <Button

       android:id="@+id/add"

       android:layout\_width="wrap\_content"

       android:layout\_height="wrap\_content"

       android:text="Додати"

       android:layout\_marginBottom="16dp"/>

   <ListView

       android:id="@+id/listView"

       android:layout\_width="match\_parent"

       android:layout\_height="wrap\_content"/>

</LinearLayout>

context\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android">

   <item

       android:id="@+id/delete"

       android:title="Видалити" />

</menu>

Note

package com.example.notessample

data class Note(

   val id: Int = 0, // ID запису

   val text: String // Текст запису

)

DataBaseHandler.kt

package com.example.notessample

import android.content.ContentValues

import android.content.Context

import android.database.sqlite.SQLiteDatabase

import android.database.sqlite.SQLiteOpenHelper

import android.widget.Toast

const val *DATABASE\_NAME* = "MyDB"

const val *TABLE\_NAME* = "Notes"

const val *COL\_NOTE* = "note"

const val *COL\_ID* = "id"

class DataBaseHandler(context: Context) : SQLiteOpenHelper(context, *DATABASE\_NAME*, null, 1) {

   private val context = context

   override fun onCreate(db: SQLiteDatabase?) {

       val createTable = "CREATE TABLE $*TABLE\_NAME* ($*COL\_ID* INTEGER PRIMARY KEY AUTOINCREMENT, $*COL\_NOTE* TEXT)"

       db?.execSQL(createTable)

   }

   override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {

       db?.execSQL("DROP TABLE IF EXISTS $*TABLE\_NAME*")

       onCreate(db)

   }

   fun insertData(note: Note) {

       val db = this.*writableDatabase*

val cv = ContentValues()

       cv.put(*COL\_NOTE*, note.text)

       val result = db.insert(*TABLE\_NAME*, null, cv)

       if (result == -1L) {

           Toast.makeText(context, "Помилка при додаванні запису", Toast.*LENGTH\_SHORT*).show()

       }

   }

   fun readData(): MutableList<Note> {

       val list = *mutableListOf*<Note>()

       val db = this.*readableDatabase*

val query = "SELECT \* FROM $*TABLE\_NAME*"

       val result = db.rawQuery(query, null)

       if (result.moveToFirst()) {

           do {

               val note = Note(

                   id = result.getInt(result.getColumnIndexOrThrow(*COL\_ID*)),

                   text = result.getString(result.getColumnIndexOrThrow(*COL\_NOTE*))

               )

               list.add(note)

           } while (result.moveToNext())

       }

       result.close()

       db.close()

       return list

   }

   fun deleteData(noteId: Int): Int {

       val db = this.*writableDatabase*

val result = db.delete(*TABLE\_NAME*, "$*COL\_ID* = ?", *arrayOf*(noteId.toString()))

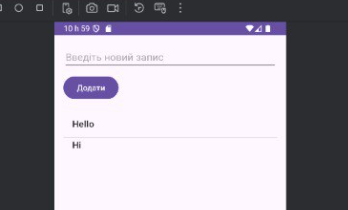
       db.close()

       return result

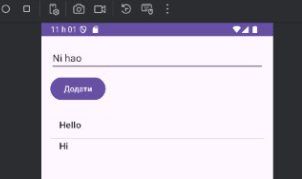
   }

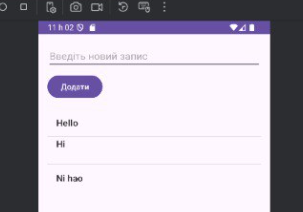
}

Результат роботи:

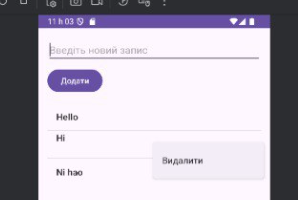


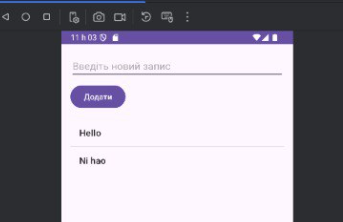
Можемо добавити новий елемент:





І аналогічно є змога видалити об’єкт:





***Висновок:*** На даній лабораторній роботі я навчилася працювати з базами даних у середовищі Android без використання спеціальних класів-адаптерів і з ними. Також засвоїла принципи створення бази даних, виконання SQL-запитів для додавання, видалення та оновлення даних

***Контрольні запитання:***

1. ***Як працює база даних SQLite?***

SQLite - це проста, вбудована база даних, яка зберігає всі дані в одному файлі на пристрої. Вона не вимагає окремого сервера, що робить її легкою та зручною для мобільних додатків. Ти можеш уявити SQLite як електронну таблицю, але з більш складними можливостями для запитів та організації даних.

1. ***Чому SQLite настільки популярна?***

В SQLite є ряд переваг, що дає змогу ставити популярнішим ніж інші продукти-аналоги:

* Легко встановити та використовувати.
* Дуже швидка для невеликих баз даних.
* Не займає багато місця на пристрої.
* Досить надійна для більшості мобільних додатків.
* Вона з відкритим кодом і безкоштовна для використання.

1. ***За допомогою якого метода можна закрити БД і для чого це роблять?***

Щоб закрити базу даних, використовується метод close() на об'єкті бази даних. Роблять це для звільнення ресурсів. Коли ти більше не працюєш з базою, закриття звільняє пам'ять, яку вона займала. Також є важливим фактором забезпечення цілісності даних. Закриття бази гарантує, що всі зміни будуть збережені на диску.

1. ***Яку роботу виконує клас SimpleCursorAdapter?***

SimpleCursorAdapter - це клас в Android, який використовується для відображення даних з бази даних SQLite у ListView. Він бере дані з курсора (результату SQL запиту) і відображає їх у вигляді списку.

Наприклад ми можемо створити ContentValues для нових даних, а потім використовувати метод update для зміни запису з певним ID.

1. ***Наведіть приклад оновлення та видалення рядків з БД.***

Приклад оновлення:

fun updateNote(note: Note, newText: String) {

val db = dbHandler.writableDatabase

val values = ContentValues().apply { put(COL\_NOTE, newText) }

db.update(TABLE\_NAME, values, "\_id=?", arrayOf(note.id.toString()))

db.close()

}

Він змінює текст запису за його ID. Створили нові значення для оновлення, вказали умову (який запис міняємо) і виконали оновлення.

Приклад видалення:

fun deleteNoteById(noteId: Int) {

val db = dbHandler.writableDatabase

db.delete(TABLE\_NAME, "\_id=?", arrayOf(noteId.toString()))

db.close()

}

Видаляємо запис за його ID. Вказуємо таблицю, умову і виконаємо видалення.