Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет Інформатики та Обчислювальної Техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №5

з дисципліни «Розробка мобільних застосувань під Android»

на тему

«ДОСЛІДЖЕННЯ РОБОТИ З ВБУДОВАНИМИ ДАТЧИКАМИ»

Виконав:
студент групи ІС-21
Качалов А. М.

Викладач:
Орленко С. П.

Київ – 2025

Мета роботи: ознайомитись з можливостями вбудованих датчиків мобільних пристроїв та дослідити способи їх використання для збору та обробки даних.

ЗАВДАННЯ

БАЗОВЕ (10/20 балів). Написати програму під платформу Андроїд, яка має інтерфейс для виведення даних з обраного вбудованого датчика (тип обирається самостійно, можна відслідковувати зміни значень і з декількох датчиків).

ПОВНЕ (20/20). Функціональність базового додатку додатково розширюється обробкою отриманих даних та виведенням їх у відповідній формі. Примітка: конкретного варіанту не передбачено, студент сам обирає завдання та вигляд програми. Приклади очікуваних робіт: - «будівельний рівень» з виведенням лінії горизонту та кутом нахилу; - компас з ілюстрацією стрілки (циферблату з позначеними сторонами світу); - крокомір (підрахунок кількості кроків); - додаток для вимірювання перевантажень в авто (G-force meter); - автоматичне регулювання яскравості та екрану в залежності від рівня освітлення, але ще б додати автозаглушення екрану при піднесенні до перешкоди (до вуха під час розмови або «в кишені»), щоб уникнути ненавмисних дотиків; - барометр з прогнозом погоди (мова про опади – зміна атмосферного тиску, а, можливо, і вологості з температурою).

HARD TASK (не обов’язково). Оскільки конкретного варіанту немає, то і конкретного завдання модифікацій додатку теж немає, але слід додати або додаткову аналітику, або використовувати інші засоби для підвищення функціональності програми. На прикладі крокоміру можна додати аналітику «за день» або «за тренування»: відобразити зміни активності, спробувати обчислити пройдену відстань чи витрачені калорії, для візуалізації можна побудувати відповідні графіки. На прикладі «вимірювача перевантажень в авто» можна спробувати додатково визначити швидкість та пройдену відстань, спробувати відмалювати карту пройденого маршруту, якщо рух кільцевий – можна вести відлік часу проходження кола, або окремих його ділянок. Для визначення швидкості/дистанції можна спробувати інтегрування або через визначення місцезнаходження на основі геолокації.

P.S. Хоч камера та мікрофон не є класичними датчиками, але на їх основі теж можна виконати лабораторну роботу (виділення контурів, розпізнавання, ar і т.д.).

Хід роботи

В даній лабраторні роботі реалізуємо наступний функціонал: при наближенні об’єктів до екрану телефона він буде заглушати яскравіть.

У файлі activity_main визначено LinearLayout, в якому знаходяться два TextView: Освітленість та Наближення.

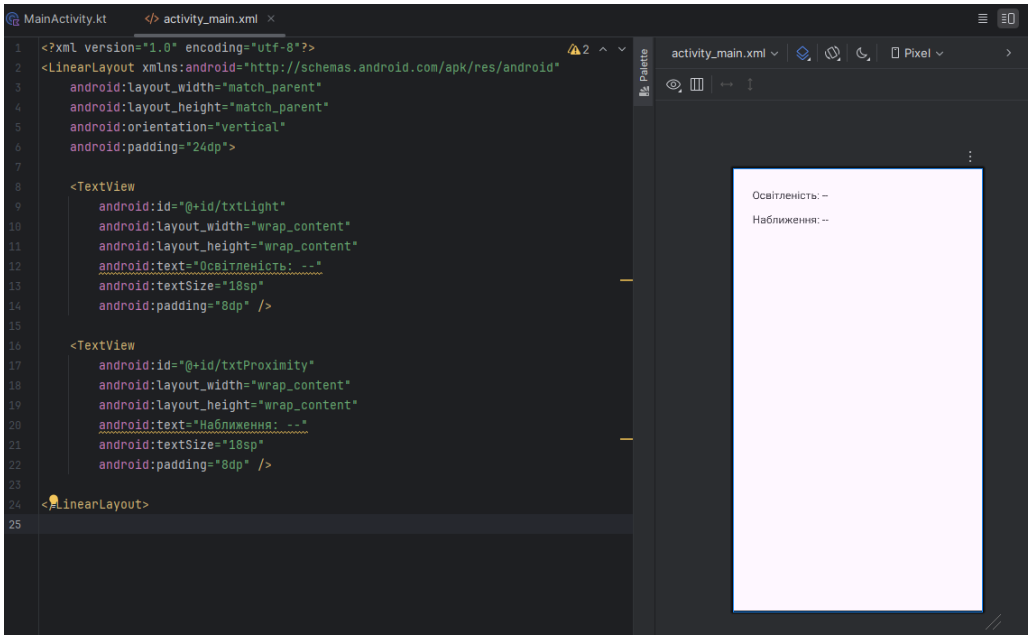

Рис. 1 – Фрагмент коду та зовнішній вигляд програмного продукту

У файлі AndroidManifest додано дозвіл (permission) для отримання можливості внесення змін у налаштування телефона через нашу програму.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.WRITE_SETTINGS"/>
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="lab5"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Lab5"
        tools:targetApi="31">
        <activity
```

Рис. 2 – Фрагмент коду файлу AndroidManifest.xml

У ході реалізації застосунку було використано два вбудованих сенсори Android-пристрою — датчик освітленості та датчик наближення. Через *SensorManager* здійснюється ініціалізація та підключення до відповідних сенсорів. Значення з них зчитуються у режимі реального часу за допомогою обробника подій *onSensorChanged*.

```
//Змінні для сенсорів
private lateinit var sensorManager: SensorManager
private var lightSensor: Sensor? = null
private var proximitySensor: Sensor? = null
```

Рис.3 – Фрагмент коду із оголошенням змінних для сенсорів

```
//Отримання інформації з сенсорів
sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
lightSensor = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT)
proximitySensor = sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY)
```

Рис. 4 – Фрагмент коду із отриманням інформації з сенсорів

Для датчика освітленості реалізовано автоматичне масштабування рівня освітленості у діапазоні від 10 до 255, після чого відповідно змінюється системна яскравість екрана (за умови наданого дозволу на зміну системних налаштувань).

```
    //Обробка освітленості
    if (Settings.System.canWrite( context: this)) {
        val minLux = 1f //Мінімум
        val maxLux = 1000f //Максимум

        // Масштабуємо освітленість у діапазон з 10 до 255
        val normalized = ((lightLevel - minLux) / (maxLux - minLux)).coerceIn(0f, 1f)
        val brightnessValue = (10 + normalized * (255 - 10)).toInt()
        //Зміна яскравості
        Settings.System.putInt(contentResolver, Settings.System.SCREEN_BRIGHTNESS, brightnessValue)
    }
}
```

Рис. 5 – Фрагмент коду із масштабування рівня освітленості

Для сенсора наближення реалізовано механізм «глушіння» екрана — при фіксації об’єкта на малій відстані до пристрою активується повноекранний чорний *View*, який блокує всі дотики, імітуючи вимкнення дисплея. Дана поведінка дозволяє моделювати функціональність, подібну до роботи смартфонів під час дзвінків.

```
            //Обробка датчика наближення
            Sensor.TYPE_PROXIMITY -> {
                val proximityValue = event.values[0]
                txtProximity.text = "Наближення: $proximityValue cm"

                val maxRange = proximitySensor?.maximumRange ?: 0f
                //Якщо близько, вимикаємо екран
                if (proximityValue < maxRange) {
                    activateBlackout( enable: true)
                } else {
                    activateBlackout( enable: false)
                }
            }
        }
    }


    //Показує або ховає чорний екран
    private fun activateBlackout(enable: Boolean) {
        if (enable && !isBlackoutActive) {
            blackoutView.visibility = View.VISIBLE
            isBlackoutActive = true
        } else if (!enable && isBlackoutActive) {
            blackoutView.visibility = View.GONE
            isBlackoutActive = false
        }
    }
```

Рис. 6 – Фрагмент коду реалізації механізму затемнення екрана

Передбачено також правильне керування роботою сенсорів — їх активація при відновленні активності (*onResume*) та деактивація при її призупиненні (*onPause*), що дозволяє ефективно використовувати ресурси пристрою.

```kotlin
//Якщо повертаємось до активності, то застосовуємо сенсори
override fun onResume() {
    super.onResume()
    lightSensor?.also {
        sensorManager.registerListener( listener: this, it, SensorManager.SENSOR_DELAY_NORMAL)
    }
    proximitySensor?.also {
        sensorManager.registerListener( listener: this, it, SensorManager.SENSOR_DELAY_NORMAL)
    }
}

//Якщо вийшли з додатку, припиняємо замірювання
override fun onPause() {
    super.onPause()
    sensorManager.unregisterListener( listener: this)
}
```

Рис. 7 – Фрагмент коду реалізації зупинки та відновлення роботи активності

Висновок

У ході виконання роботи було досліджено принципи роботи з вбудованими датчиками мобільного пристрою на платформі Android. Зокрема, реалізовано обробку даних з датчика освітленості та датчика наближення, що дозволило на практиці ознайомитись зі способами зчитування та використання сенсорної інформації. Було отримано навички налаштування сенсорів, опрацювання їхніх подій та динамічного реагування інтерфейсу на зміни середовища. Отримані результати підтверджують широкі можливості вбудованих сенсорів для створення адаптивних, чутливих до контексту мобільних застосунків.

Контрольні питання

1. Приклади вбудованих датчиків та величини, які з них можна зчитати

У мобільних пристроях на базі Android доступна велика кількість вбудованих сенсорів, що дозволяють взаємодіяти з фізичним середовищем. Серед них можна виділити:

I. Акселерометр — вимірює прискорення вздовж трьох осей (X, Y, Z), що дозволяє визначити положення пристрою в просторі або фіксувати рух.
II. Гіроскоп — фіксує кутові швидкості обертання, що дозволяє точніше визначати обертання пристрою.
III. Магнітометр (компас) — визначає орієнтацію відносно магнітного поля Землі, використовується для навігації.
IV. Датчик освітлення — дозволяє зчитувати рівень навколишнього освітлення для автоматичного регулювання яскравості екрану.
V. Барометр — вимірює атмосферний тиск, що може використовуватись для визначення висоти над рівнем моря.
VI. Датчик наближення — виявляє наявність об'єкта біля екрана (наприклад, під час розмови по телефону).
VII. Датчик температури та вологості — деякі пристрої мають сенсори для вимірювання кліматичних умов навколо користувача.

2. Особливості роботи з вбудованими датчиками

Робота з вбудованими датчиками на Android передбачає низку технічних та архітектурних особливостей. Основними з них є:

I. Використання SensorManager — усі операції з датчиками починаються з отримання доступу до системної служби SensorManager, через яку здійснюється підключення до необхідного сенсора.

II. Обробка даних у реальному часі — дані від сенсорів надходять асинхронно через обробник подій SensorEventListener, що дозволяє ефективно реагувати на зміни без затримок.

III. Оптимізація споживання енергії — слід обирати відповідну частоту оновлення даних (наприклад, SENSOR_DELAY_NORMAL, FASTEST, тощо), оскільки надто часте зчитування може швидко розряджати акумулятор.

IV. Комбінування сенсорів — для більш точного аналізу часто використовуються комбінації даних з кількох сенсорів (наприклад, акселерометр + гіроскоп для побудови стабільної моделі руху).

V. Обмеження вірогідності — дані з сенсорів можуть містити шум або похибки, тому в критичних застосунках важливо застосовувати фільтрацію чи корекцію сигналу.

VI. Життєвий цикл — датчики мають бути активовані лише тоді, коли вони дійсно потрібні (наприклад, у onResume) та відключені у onPause або onStop для економії ресурсів.