ПРАТКИЧНА РОБОТА № 7.
РОБОТА З ОБ'ЄКТАМИ ACTIVITY ТА INTENT

Мета: метою виконання лабораторної роботи є набуття студентами навичок:

- створення нової Activity в середовищі Android Studio;
- оголошення батьківської та дочірньої Activity для навігації;
- запуску на виконання Activity з допомогою явного Intent;
- передачі даних між Activity.

Короткі теоретичні відомості

Activity представляє собою окремий екран, за допомогою користувач додатку може виконати єдине цілеспрямоване завдання, наприклад, зробити фотографію, надіслати електронний лист або переглянути карту. Activity зазвичай представляється користувачеві як повноекранне вікно.

Додаток зазвичай складається з декількох екранів, які слабко пов'язані один з одним. Кожен екран додатку – це окрема Activity. Зазвичай одна Activity у додатку визначається як основна (*MainActivity.java*), яка подається користувачеві під час запуску програми. Потім основна Activity може запускати інші для виконання різних дій.

Кожного разу, коли запускається нова Activity, попередня Activity зупиняється, але система зберігає Activity у стеку (*back stack*). Коли нова Activity запускається, вона поміщується на вершину стеку і отримує фокус користувача. Коли користувач закінчує роботу поточної Activity натискаючи на кнопку Back, ця Activity вилучається зі стеку та знищується, а робота попередньої Activity відновлюється.

Activity запускається або активується з допомогою Intent. Об'єкт Intent – це асинхронне повідомлення, яке ви можете використовувати у своїй Activity, щоб надіслати запит на дію з іншої Activity чи з іншого компонента програми. Intent використовує для запуску однієї Activity з іншої Activity та для передачі даних між ними. Intent може бути явним або неявним:

- явний (*explicit*) Intent – це такий Intent, мета якого є відомою. Тобто ви знаєте повністю назву класу конкретної Activity. Ви визначаєте конкретний цільовий компонент для передачі даних;
- неявний (*implicit*) Intent – це такий Intent, для якого ім'я цільового компонента є невідомим, але у вас є певна загальна для виконання дія. Для такого Intent визначається функціональність, але не цільовий компонент.

У лабораторній роботі використовуються явні об’єкти типу Intent.

Кожна нова Activity, яку додають до проекту, має власний макет (*layout*) та Java-файл, відокремлені від інших в проекті. У файлі *AndroidManifest.xml* вона має свій відповідний елемент <*activity*>. Як і головна Activity проекту, всі інші Activity успадковуються від класу AppCompatActivity.

Хоча кожна Activity в проекті слабо пов’язана з іншими, можна визначити Activity як частину іншої Activity в файлі *AndroidManifest.xml*. Це відношення батько-нащадок змушує Android додавати навігаційні підказки, такі як стрілки «ліворуч» у рядку заголовка для кожного Activity.

Об’єкт Intent може передавати дані цільовому Activity двома способами: використовуючи поле даних (*intent data*) або додатки (*intent extras*). Поле даних являє собою URI, який вказує на конкретні дані для дії. *Intent extras* є об’єктом типу *Bundle*, що представляє собою колекцію ключ/значення.

Хід роботи

1. Створення проекту, який складається із двох Activity

1.1 Створення нового проекту

Запустіть Android Studio та створіть новий проект під назвою *Two Activities*. Серед шаблонів Activity виберіть *Empty Activity*. Залишить ім’я головного Activity по замовчуванню (*MainActivity*) без змін.

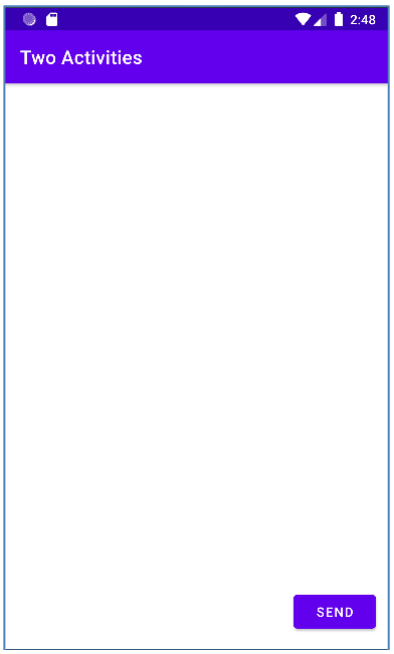
1.2 Визначення Layout для головного Activity

1.2.1 Відкрийте в редакторі розмітки *res>layout>activity_main.xml*.

1.2.2 Перейдіть в закладку *Design* і видаліть *TextView*, який містить напис «Hello World».

1.2.3 При увімкненій по замовчуванню опції *Autoconnect* перетягніть кнопку *Button* із панелі компонентів у правий нижній кут розмітки інтерфейсу користувача.

1.2.4 В панелі атрибутів *Attributes* встановіть значення її властивостей: *ID* на *button_main, layout_width* та *layout_height* на *wrap_content,* напис *Send*. В результаті повинні отримати:



1.2.5 Перейдіть на закладу *Text* для редагування xml-коду. Додайте для кнопки атрибут: `android:onClick="launchSecondActivity"`

1.2.6 Застосуйте ресурси, щоб присвоїти кнопці напис *Send*. Відповідному ресурсу присвойте ім'я *button_main*.

```
 android:text="@string/button_main"
//файл activity_main.xml
<string name="button_main">Send</string>
//файл strings.xml
```

Розмітка кнопки повинна мати вигляд:

```
<Button
    android:id="@+id/button_main"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginRight="16dp"
    android:text="@string/button_main"
    android:onClick="launchSecondActivity"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintRight_toRightOf="parent"/>
```

1.3 Прив'язування обробника подій натискання мишкою до кнопки

Визначимо метод *launchSecondActivity*() та прив'яжемо його до атрибуту *android:onClick* нашої кнопки.

1.3.1 Натисніть на *launchSecondActivity()* в коді файлу

activity_main.xml.

1.3.2 Натисніть Alt+Enter та виберіть команду *Create launchSecondActivity(View) in MainActivity* в меню.

1.3.3 Всередині коду методу *launchSecondActivity()* напишіть код:

```
Log.d(LOG_TAG, "Button clicked!");
```

1.3.4 У верхній частині класу MainActivity додайте константу для змінної LOG-TAG:

```
private static final String LOG_TAG =
MainActivity.class.getSimpleName();
```

1.3.5 Запустіть проект на виконання. При натисканні на кнопку *Send* ви повинні побачити повідомлення "*Button Clicked*!" у вікні логів *Logcat*. Код класу MainActivity тепер повинен виглядати так:

```
package com.example.android.twoactivities;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;

public class MainActivity extends AppCompatActivity {
    private static final String LOG_TAG =
    MainActivity.class.getSimpleName();

    @Override protected void onCreate(Bundle
    savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void launchSecondActivity(View view) {
        Log.d(LOG_TAG, "Button clicked!");
    }
}
```

2. Створення та запуск другого Activity

2.1 Створення другого Activity

2.1.1 Натисніть на папку *app* у проекті та виберіть команду *File > New>Activity>Empty Activity*.

2.1.2 Назвіть другу Activity ім’ям SecondActivity. Переконайтесь, що

опції *Generate Layout File* та *Backwards Compatibility (AppCompat)* залишились відміченими. Ім'я файлу розмітки повинне бути *activity_second*. Не відмічайте опцію *Launcher Activity*.

2.1.3 Натисніть *Finish*. Android Studio згенерує та додасть в проект файли, які відповідають другому Activity, а також внесе необхідні зміни у файл маніфесту проекту.

2.2 Модифікація файла маніфесту

2.2.1 Відкрийте файл *manifests* > *AndroidManifest.xml*.

2.2.2 Знайдіть елемент <*activity*>, який відповідає другому Activity

```
<activity
     android:name=".SecondActivity">
     . . .
</activity>
```

2.2.3 Перепишіть цей елемент таким вмістом:

```
<activity
     android:name=".SecondActivity"
     android:label = "Second Activity"
     android:parentActivityName=".MainActivity"/>
```

Атрибут *label* тут задає ім'я Activity, яке буде виведено на панель. За допомогою атрибуту *parentActivityName* ми вказуємо, що батьківською по відношенню до Second Activity буде MainActivity.

2.3 Визначення розмітки для другої Activity

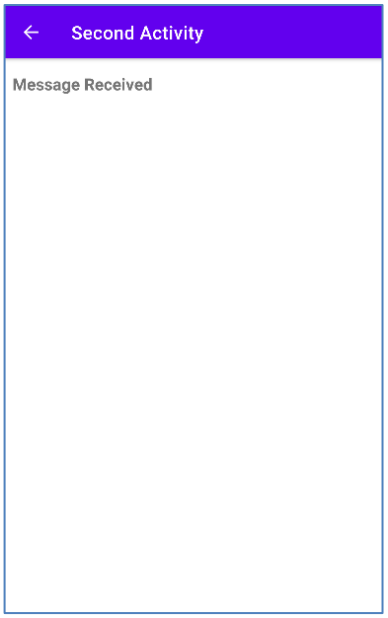
2.3.1 Відкрийте файл *activity_second.xml* та перейдіть в режим дизайнера (закладка *Design*)

2.3.2 Перетягніть *TextView* з палітри компонентів у верхній лівий кут розмітки та задайте прив'язку даного елемента до верхнього та лівого краю розмітки. Встановіть атрибути *TextView* в такі значення:

Атрибут	**Значення**
id	*text_header*
Top margin	*16*
Left margin	*8*
layout_width	*wrap_content*
layout_height	*wrap_content*

text	*Message Received*
textAppearance	*AppCompat.Medium*
textStyle	*B (bold)*

Значенням атрибуту *textAppearance* є найменування стандартної теми Android, яка визначає основні стилі шрифту. Розмітка Activity повинна зараз виглядати так:



2.3.3 Перейдіть в режим xml-розмітки (закладка *Text*) та зробіть так, щоб стрічка *Message Received* вибиралася із ресурсу під назвою *text_header*.

```
 android:text="@string/text_header"
//файл activity_second.xml
<string name="text_header">Message Received</string>
//файл strings.xml
```

2.3.4 Додайте до *TextView* атрибут *android:layout_marginLeft="8dp"* щоб доповнити атрибут *layout_marginStart* для старіших версій Android. Тепер файл *activity_second.xml* повинен мати такий вигляд:

```
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SecondActivity">
```

```
    <TextView
        android:id="@+id/text_header"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="16dp"
        android:text="@string/text_header"
        android:textAppearance=
       "@style/TextAppearance.AppCompat.Medium"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

2.4 Додавання Intent до головної Activity проекту

2.4.1 Перейдіть у файл Java-коду головної Activity

2.4.2 Створіть Intent в методі *launchSecondActivity()*

Конструктор explicit Intent приймає два аргументи: контекст програми та конкретний компонент, для якого цей Intent призначений:

```
Intent intent = new Intent(this, SecondActivity.class);
```

2.4.3 Викликаємо метод *startActivity(),* якому в якості параметра передаємо створений intent:

```
startActivity(intent);
```

2.4.4 Запустіть проект на виконання. Тепер при натисканні на кнопку *Send*, MainActivity посилає Intent та операційна система запускає SecondActivity, яка з'являється на екрані. Щоб повернутися до головної Activity слід натиснути кнопку *Up* (кнопка ліворуч на панелі додатку вгорі) або кнопку *Back* внизу екрану.

3. Передача даних від MainActivity до SecondActivity

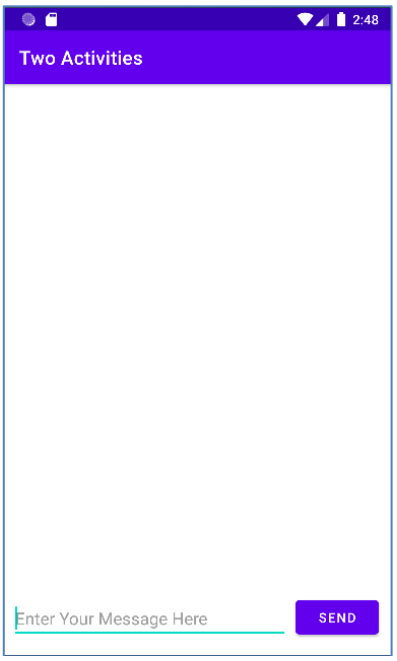
3.1 Додавання EditText до розмітки головної Activity

3.1.1 Перейдіть у файл *activity_main.xml*.

3.1.2 Перетягніть елемент *Plain Text (EditText)* з палітри у низ розмітки та задайте прив'язку до лівої сторони та до низу розмітки, а також до лівої сторони кнопки *Send*. Встановіть значення атрибутів:

Атрибут	Значення
id	*editText_main*
Right margin	*8*
Left margin	*8*
Bottom margin	*16*
layout_width	*match_constraint*
layout_height	*wrap_content*
inputType	*textLongMessage*
hint	*Enter Your Message Here*
text	*(Delete any text in this field)*

Розмітка головної Activity повинна зараз виглядати так:



Перейдіть в режим xml-розмітки та зробіть так, щоб стрічка «*Enter Your Message Here*» вибиралася з ресурсу під назвою *editText_main*.

```
<string name="editText_main">Enter Your Message Here</string>
```

Тепер файл *activity_ main.xml* повинен мати такий вигляд:

```
<androidx.constraintlayout.widget.ConstraintLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    tools:context=".SecondActivity">

    <TextView
        android:id="@+id/text_header"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="16dp"
        android:text="@string/text_header"
        android:textAppearance=
        "@style/TextAppearance.AppCompat.Medium"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/text_message"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:textAppearance=
        "@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf=
       "@+id/text_header" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

3.2 Внесення стрічки в Intent extras

3.2.1 Відкрийте файл MainActivity.java

3.2.2 Додайте константу:

```
public static final String EXTRA_MESSAGE =
"com.example.android.twoactivities.extra.MESSAGE";
```

3.2.3 Додайте приватну змінну для роботи із *EditText:*

```
private EditText mMessageEditText;
```

3.2.4 В методі *onCreate*() використайте метод *findViewById*() щоб отримати посилання на *EditText*:

```
mMessageEditText=findViewById(R.id.editText_main);
```

3.2.5 В методі *launchSecondActivity*() після оголошення Intent отримуємо текст *EditText* як стрічку:

```
String message=mMessageEditText.getText().toString();
```

3.2.6 Додаємо отриману стрічку в Inetnt, використовуючи константу *EXTRA_MESSAGE* в якості ключа:

```
intent.putExtra(EXTRA_MESSAGE, message);
```

Тепер методи повинні мати вигляд:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    mMessageEditText=findViewById(R.id.editText_main);
}

public void launchSecondActivity(View view) {
    Log.d(LOG_TAG, "Button clicked!");
    Intent intent=new Intent(this, SecondActivity.class);

    String message=mMessageEditText.getText().toString();
    intent.putExtra(EXTRA_MESSAGE, message);
    startActivity(intent);
}
```

3.3 Додавання TextView до SecondActivity

3.3.1 Перейдіть у файл activity_second.xml.

3.3.2 Додайте ще один *TextView* до розмітки після *text_header TextView* та зробіть прив’язку його до лівої сторони розмітки та низу *text_header*.

3.3.3 Встановіть значення атрибутів:

Атрибут	Значення
id	*text_message*
Top margin	*8*
Left margin	*8*
layout_width	*wrap_content*
layout_height	*wrap_content*
text	*(Delete any text in this field)*
textAppearance	*AppCompat.Medium*

Файл *activity_second.xml* повинен виглядати так:

```
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SecondActivity">

    <TextView
        android:id="@+id/text_header"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="16dp"
        android:text="@string/text_header"
        android:textAppearance=
       "@style/TextAppearance.AppCompat.Medium"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/text_message"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:textAppearance=
        "@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/text_header" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

3.4 Модифікація SecondActivity для того, щоб він міг отримати дані та відобразити їх на екрані

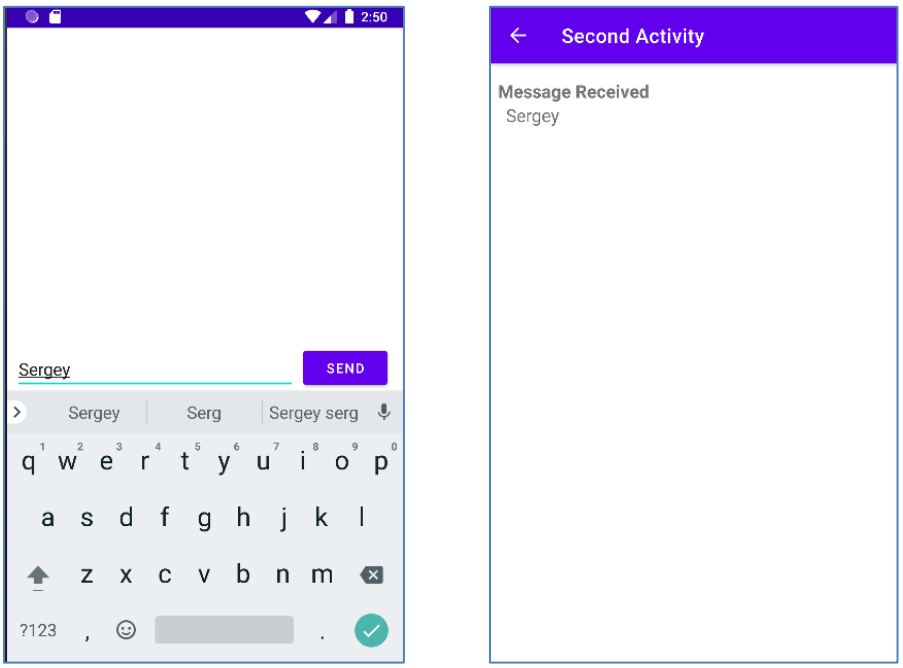
3.4.1 Відкрийте SecondActivity

3.4.2 В код методу *onCreate*() додайте наступний код:

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);

        // отримати Intent, який активує дане Activity
        Intent intent = getIntent();
        // отримати стрічку з intent по ключу EXTRA_MESSAGE
        String message =
        intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
        // використати findViewById() щоб отримати посилання на
        //textView з розмітки
        TextView textView = findViewById(R.id.text_message);
        // встановити текст для textView
        textView.setText(message);
    }
```

3.4.3 Запустіть додаток на виконання. Тепер коли ми вводимо в MainActivity повідомлення та натискаємо *Send*, запускається друга Activity, яка це повідомлення приймає і відображає на екрані:



Контрольні запитання

1. Де знаходяться в проекті файли класів activity та xml-файлирозмітки інтерфейсу?
2. Що таке Activity?
3. Опишіть життєвий цикл Activity.
4. Як можна запустити Activity?

5. Яким чином можна передавати/отримувати дані між різними Activity?
6. Що таке Intent та для чого він використовується?
7. Які бувають види Intent?

Перелік посилань

1. Гриффитс Дэвид, Гриффитс Дон. Head First. Программирование для Android. СПб.: Питер, 2018. 912 с.
2. https://codelabs.developers.google.com/codelabs/android-training-create-an-activity/index.html?index=..%2F..%2Fandroid-training#0
3. https://developer.android.com/guide/components/fundamentals.html
4. https://developer.android.com/guide/components/activities.html
5. https://developer.android.com/guide/components/intents-filters.html
6. https://developer.android.com/guide/navigation/navigation-design-graph
7. https://google-developer-training.github.io/android-developer-fundamentals-course-concepts-v2/unit-1-get-started/lesson-2-activities-and-intents/2-1-c-activities-and-intents/2-1-c-activities-and-intents.html