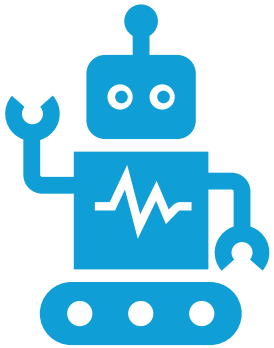

# Android Studio i JAVA

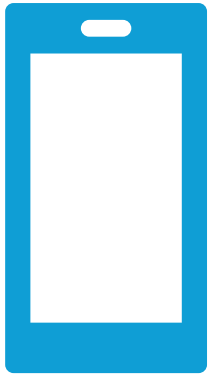
Zbigniew Kluczkowski

# Spis treści

# Wprowadzenie



Android Studio to oficjalne zintegrowane środowisko programistyczne (IDE) dla systemu Android.



Pokażę, jak poprawnie zainstalować i skonfigurować Android Studio na systemie Windows 11.

# Wymagania systemowe

Minimalne wymagania sprzętowe:

- Procesor: Intel Core i5 lub wyższy
- RAM: 8 GB (zalecane 16 GB)
- Dysk: 2 GB wolnego miejsca
- System operacyjny: Windows 11 (64-bitowy)

Oprogramowanie:

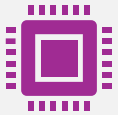
- Java Development Kit (JDK)
- Android Studio Installer

# Instalacja Android Studio

Krok 1: Uruchom pobrany plik instalacyjny (.exe).

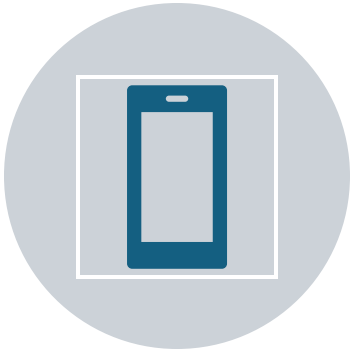
Krok 2: Postępuj zgodnie z kreatorem instalacji:

- Wybierz ścieżkę instalacji (domyślnie: C:\Program Files\Android\Android Studio)



- Zainstaluj komponenty takie jak Android SDK, AVD oraz Emulator.

# Konfiguracja Android SDK



KROK 1: PRZEJDŹ DO FILE > SETTINGS > SYSTEM SETTINGS > ANDROID SDK.

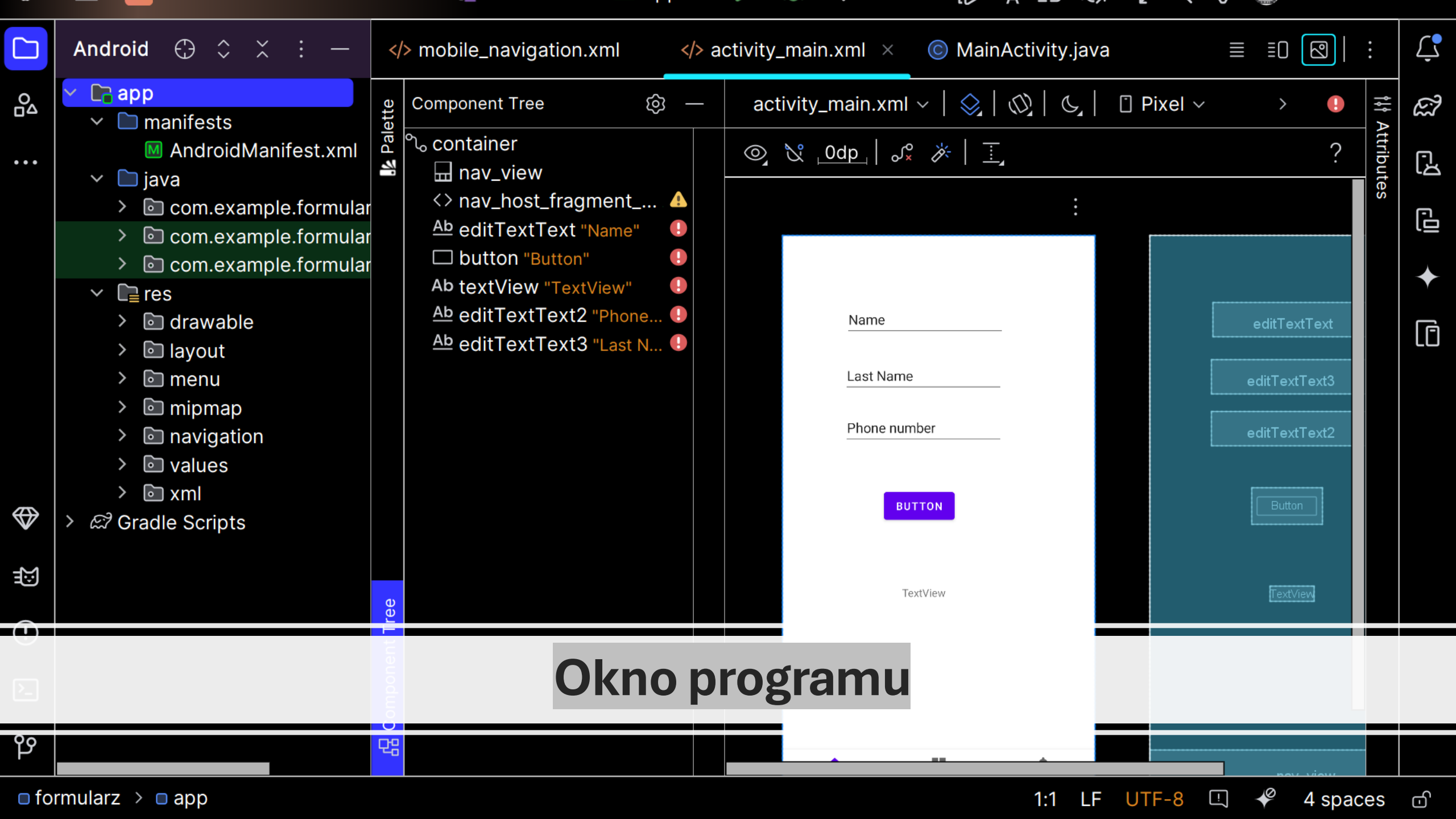
KROK 2: WYBIERZ WERSJĘ SDK, NP. ANDROID 12 (API LEVEL 31).

Problem: Emulator działa powoli.

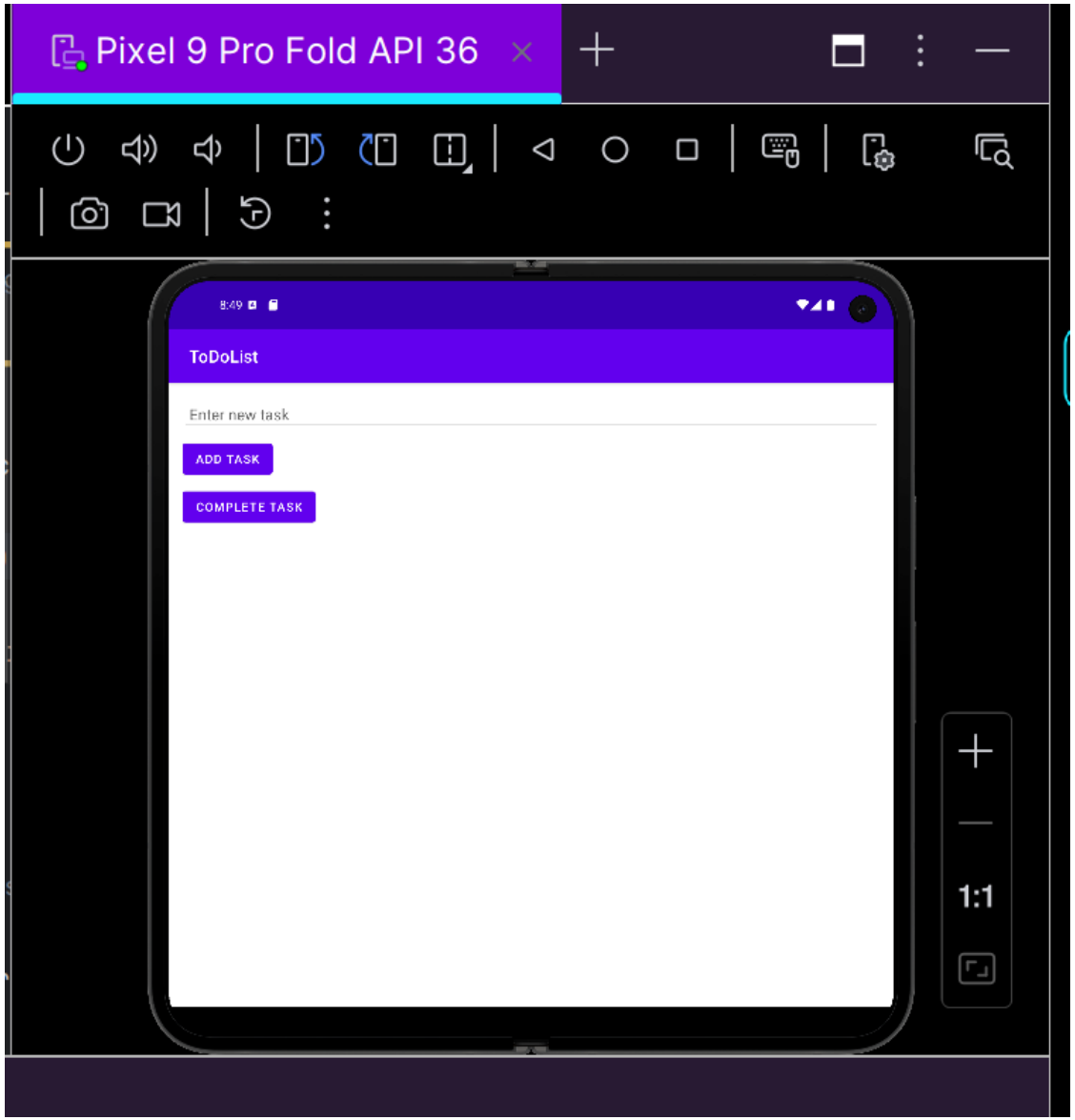
- Rozwiązanie: Włącz akcelerację HAXM.

- Rozwiązanie: Sprawdź połączenie USB i

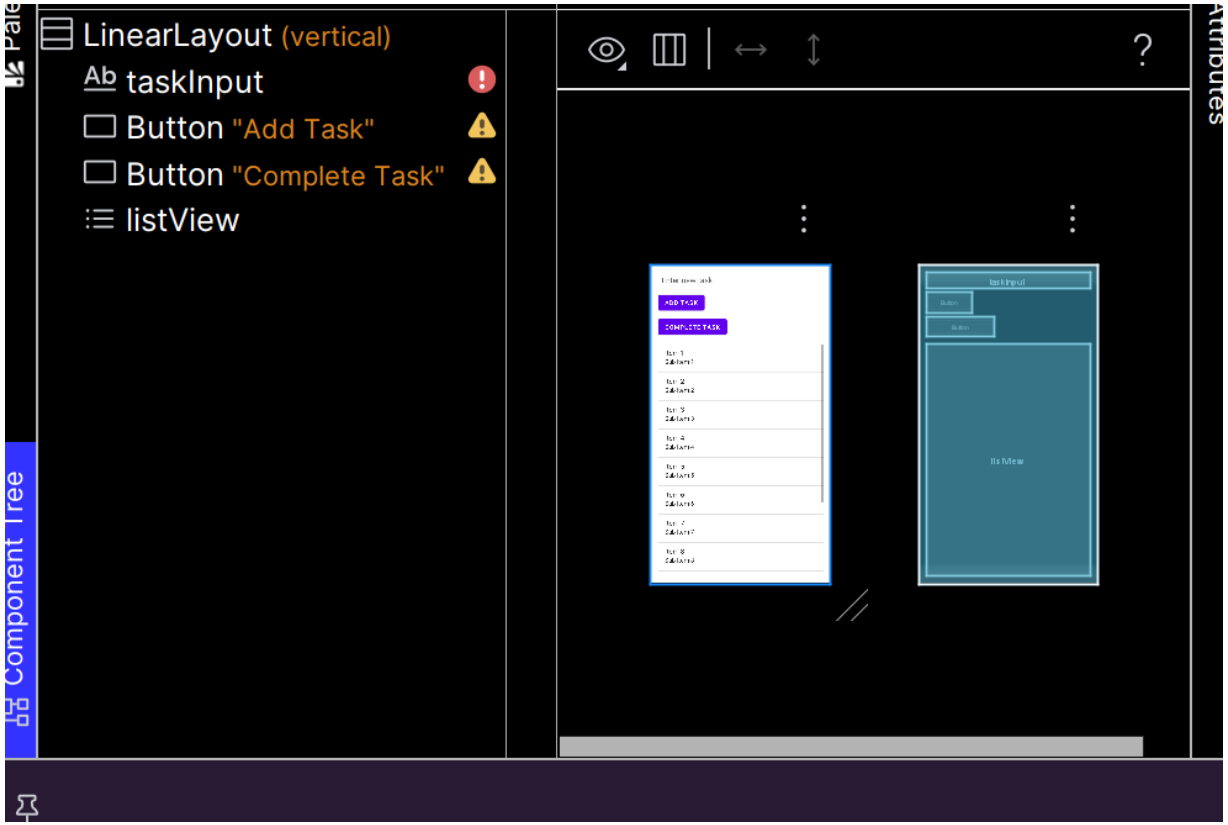
sterowniki.

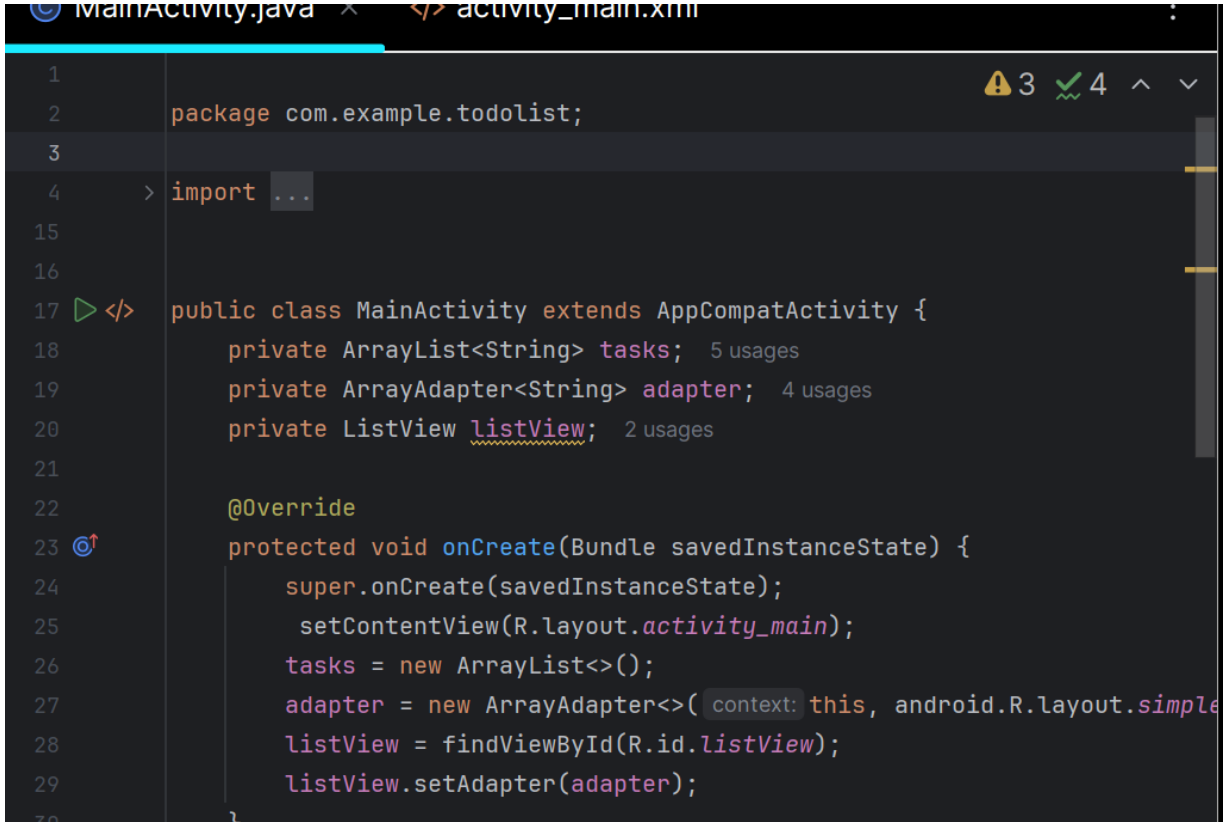

Android
app
manifests
AndroidManifest.xml
java
com.example.formular
com.example.formular
com.example.formular
res
drawable
layout
menu
mipmap
navigation
values
xml
Gradle Scripts
mobile_navigation.xml
activity_main.xml
MainActivity.java
Palette
Component Tree
container
nav_view
nav_host_fragment_...
editTextText "Name"
button "Button"
textView "TextView"
editTextText2 "Phone...
editTextText3 "Last N...
activity_main.xml
Pixel
0dp
Name
Last Name
Phone number
BUTTON
TextView
editTextText
editTextText3
editTextText2
Button
TextView
Attributes
Component Tree
Okno programu
formularz
app
1:1
LF
UTF-8
4 spaces

Emulator z włączoną aplikacją



# Dwa najważniejsze pliki



# XML – pliki opisowe interfejsu

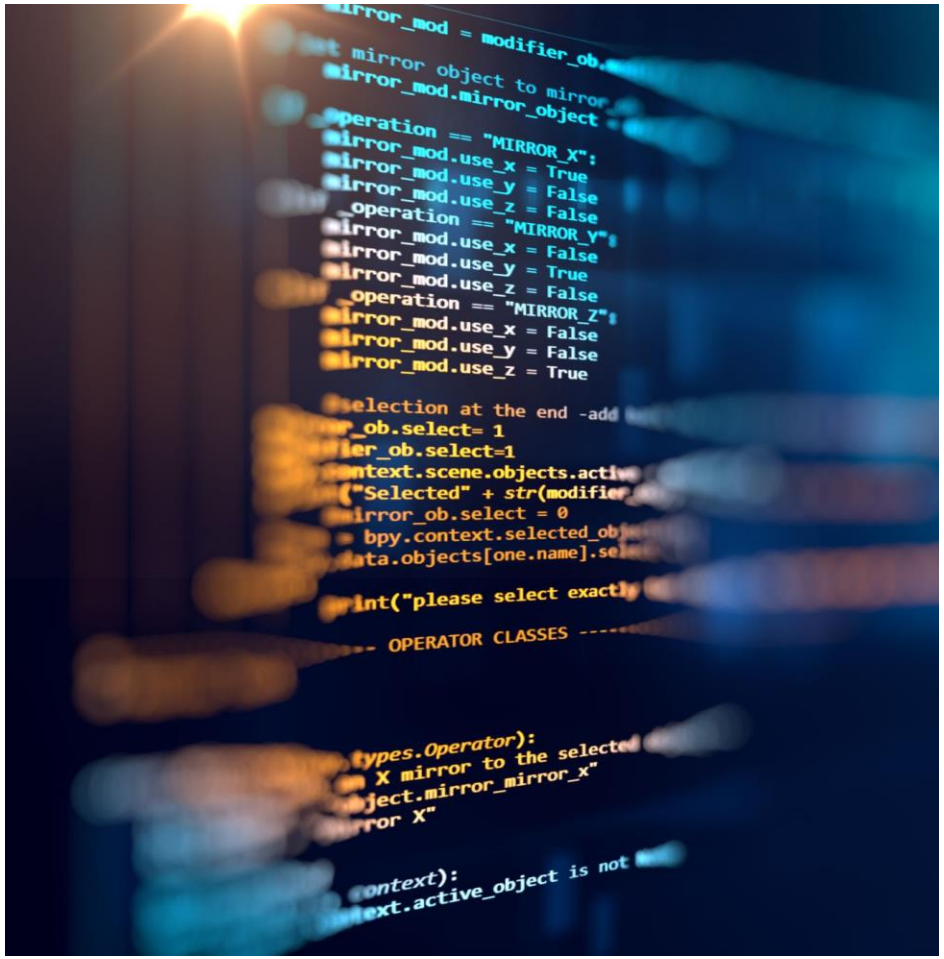
**Pliki XML można porównać do plików CSS w przypadku tworzenia stron internetowych.**

**Zawierają zwykle:**

- **Nazwę elementu**
- **Styl formatowania**
- **Miejsce elementu w siatce**
- **Przypisanie akcji do elementu.**

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
<item
 android:id="@+id/menu_settings"
android:title="Ustawienia"
 android:icon="@drawable/ic_settings"
android:showAsAction="ifRoom" />
<item
android:id="@+id/menu_about"
android:title="O aplikacji"
 android:icon="@drawable/ic_info"
  android:showAsAction="ifRoom" />
<item
 android:id="@+id/menu_exit"
android:title="Wyjście"
 android:icon="@drawable/ic_exit"
 android:showAsAction="never" />
</menu>
```

# Layout w XML




Layout w XML to kontener, który organizuje inne elementy UI, takie jak przyciski, etykiety i pola tekstowe.

**Właściwości Layout:**

- **Orientation**: Określa kierunek, w którym układ ma rozmieszczać swoje dzieci (dotyczy np. StackLayout).

  Przykład: Orientation="Vertical" (pionowy układ elementów).

- **Padding**: Ustawia odstępy wewnątrz układu od jego granic.

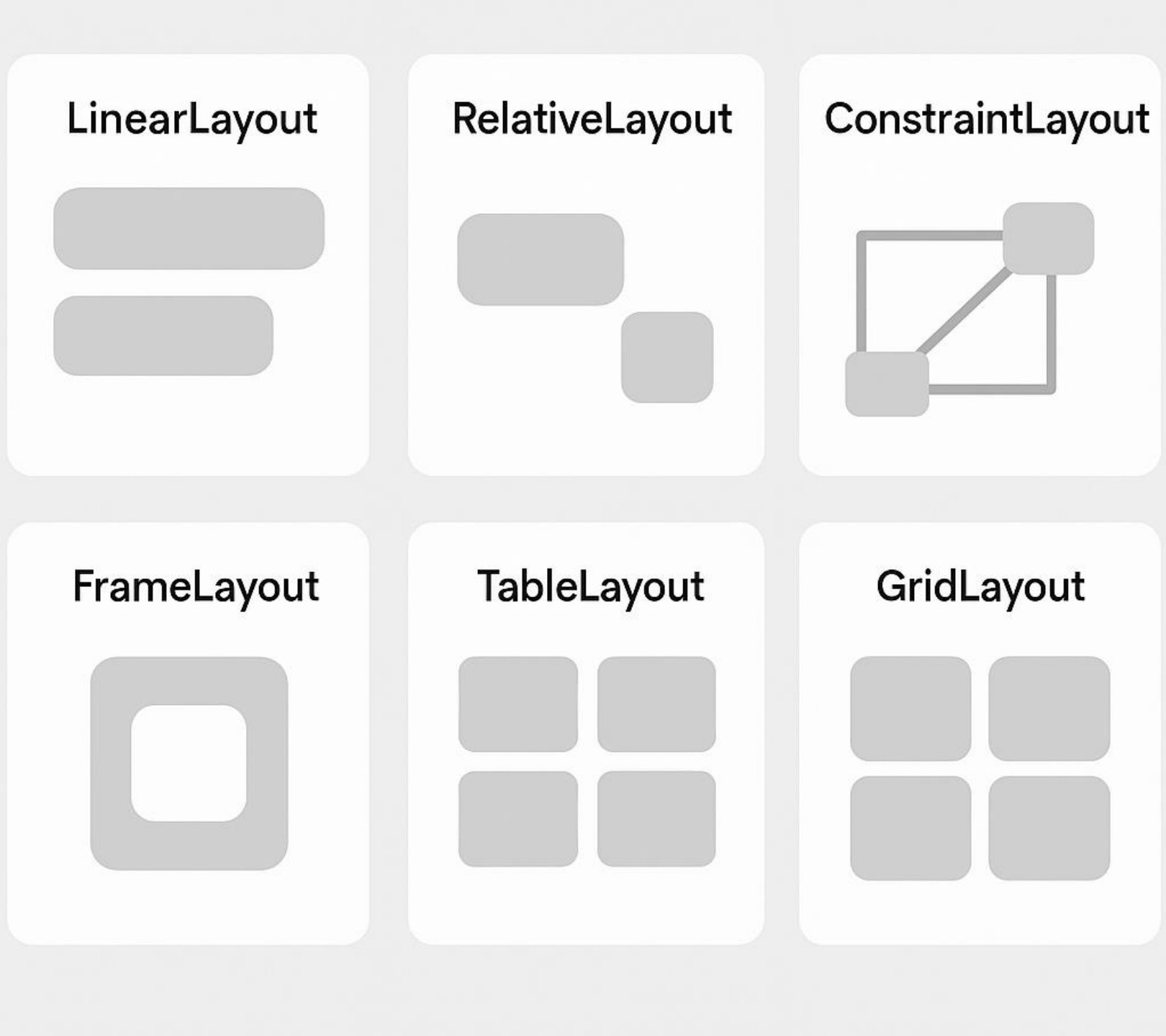
  Przykład: Padding="10".

- **Margin**: Definiuje odstępy zewnętrzne między układem a sąsiadującymi elementami.

  Przykład: Margin="5,10,5,10".

- **HorizontalOptions / VerticalOptions**: Określa pozycję dzieci w układzie.

  Przykład: HorizontalOptions="Center" (wyśrodkowanie w poziomie).

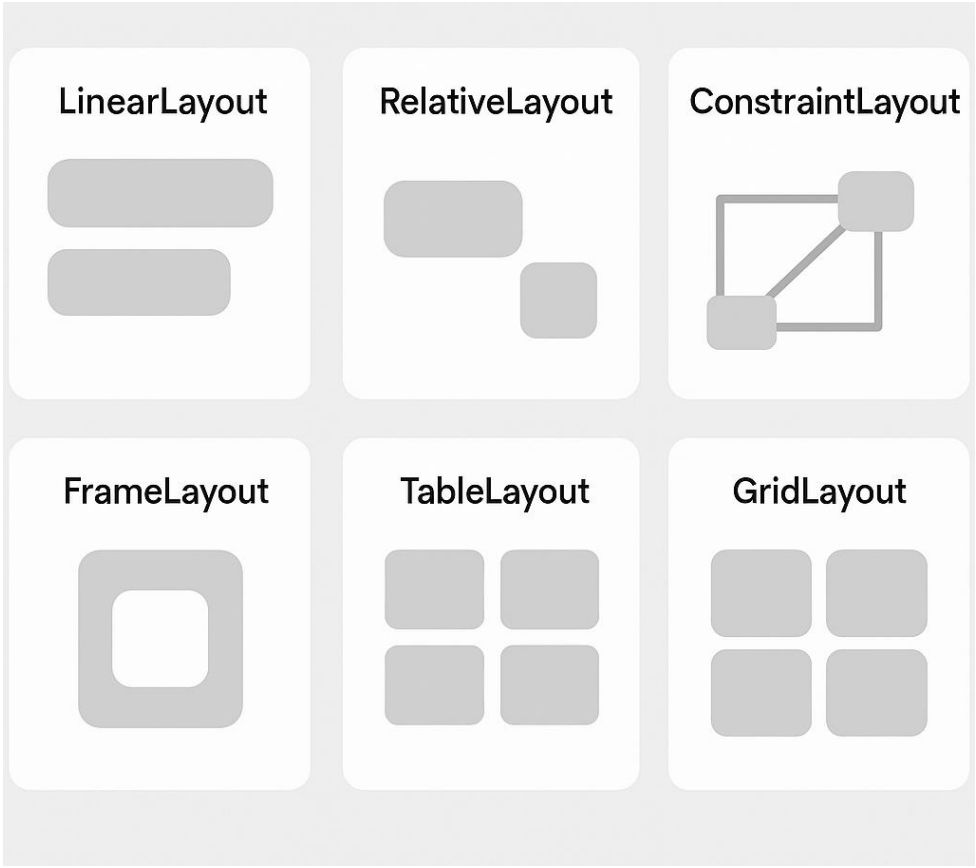



# Rodzaje layout

# Linear Layout

Ustawia elementy **jeden po drugim**, **pionowo lub poziomo**.
Atrybut kluczowy: android:orientation="vertical" lub „horizontal”.

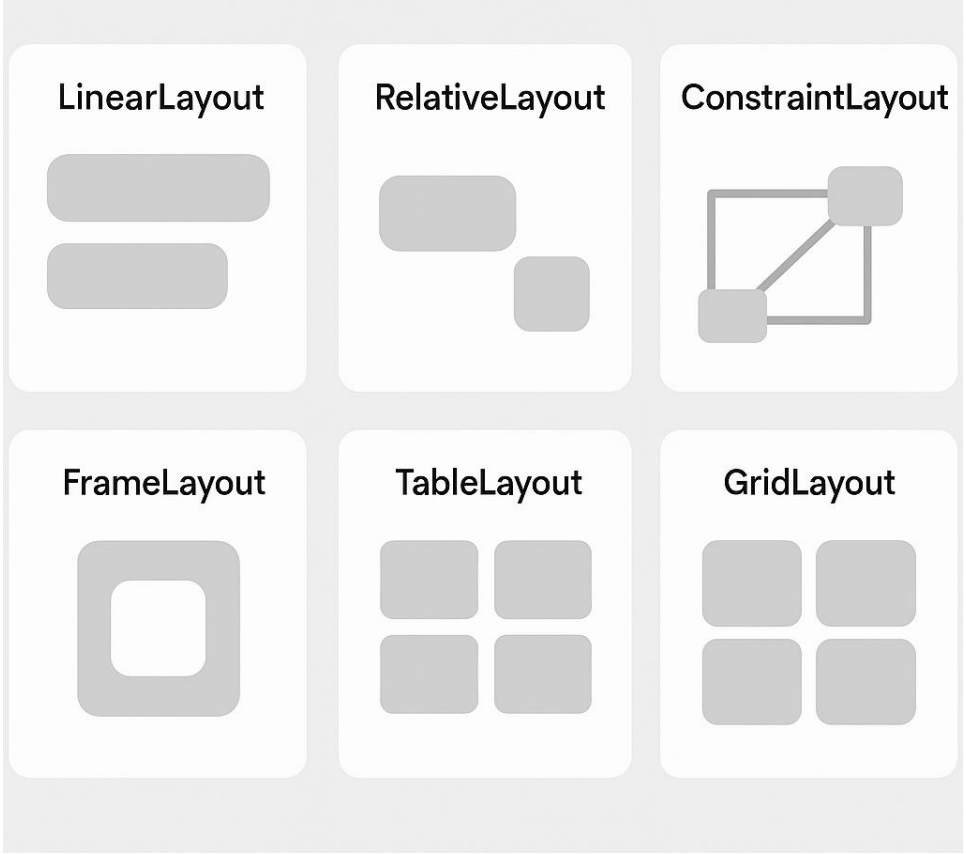
```
<LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
            <TextView android:text="Przykład" />
            <Button android:text="Kliknij mnie"
/></LinearLayout>
```



# Relative Layout

Umożliwia **pozycjonowanie elementów względem siebie** lub rodzica.

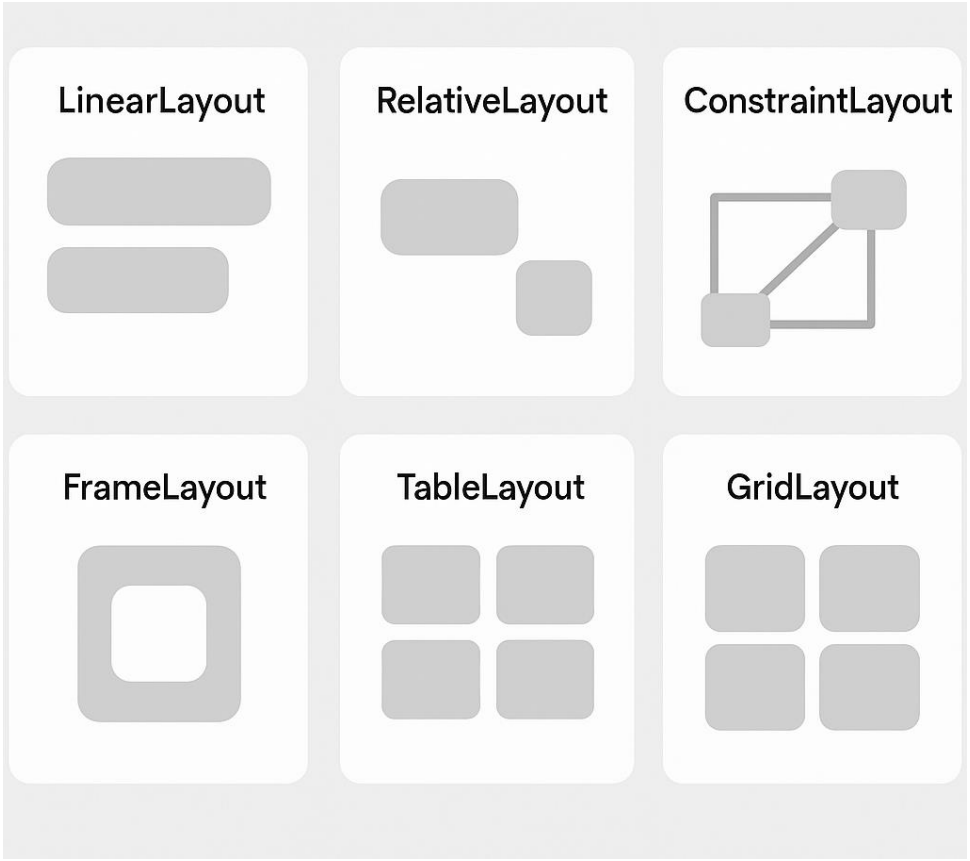
```
<RelativeLayout>
<TextView
        android:id="@+id/text1"
        android:layout_alignParentTop="true" />
<Button
        android:layout_below="@id/text1"
        android:layout_alignParentEnd="true" />
</RelativeLayout>
```



# Constraint Layout

Elastyczny układ umożliwiający **tworzenie złożonych interfejsów bez zagnieżdżania**.
Używa tzw. **constraintów** (ograniczeń) do rozmieszczenia elementów.

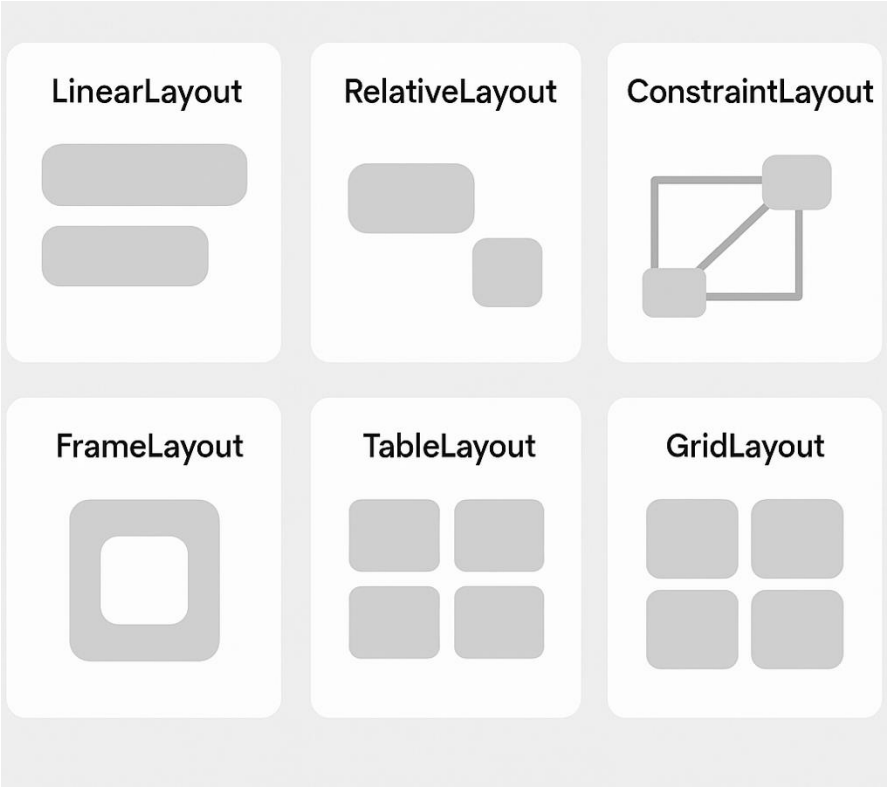
```
<ConstraintLayout>
        <TextView
                android:id="@+id/text1"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintStart_toStartOf="parent" />
        <Button
                app:layout_constraintTop_toBottomOf="@id/text1"
                app:layout_constraintStart_toStartOf="parent" />
</ConstraintLayout>
```



# Frame Layout

Wszystkie elementy są umieszczane **jeden na drugim (nakładane)** – używany często np. do fragmentów.

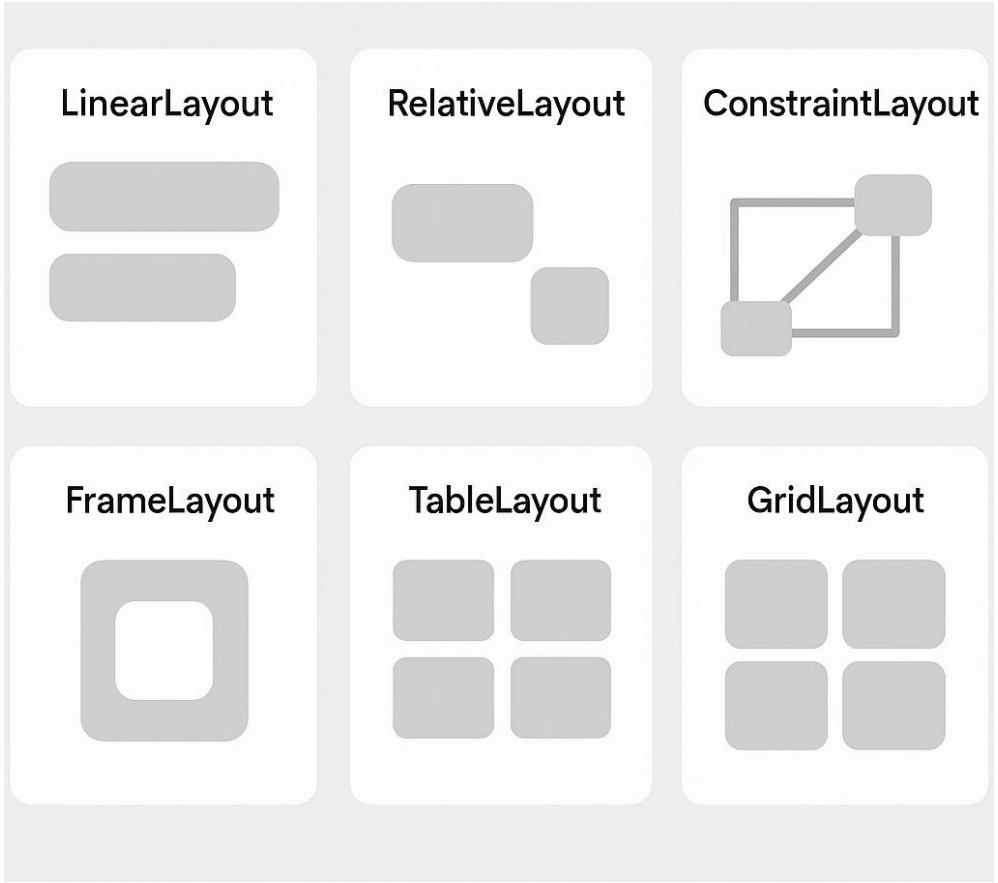
```
<FrameLayout>
        <ImageView android:src="@drawable/tlo" />
        <TextView android:text="Na tle obrazu" />
</FrameLayout>
```



# Table Layout

Układa elementy w **wierszach i kolumnach** jak tabela.
Składa się z TableRow elementów.

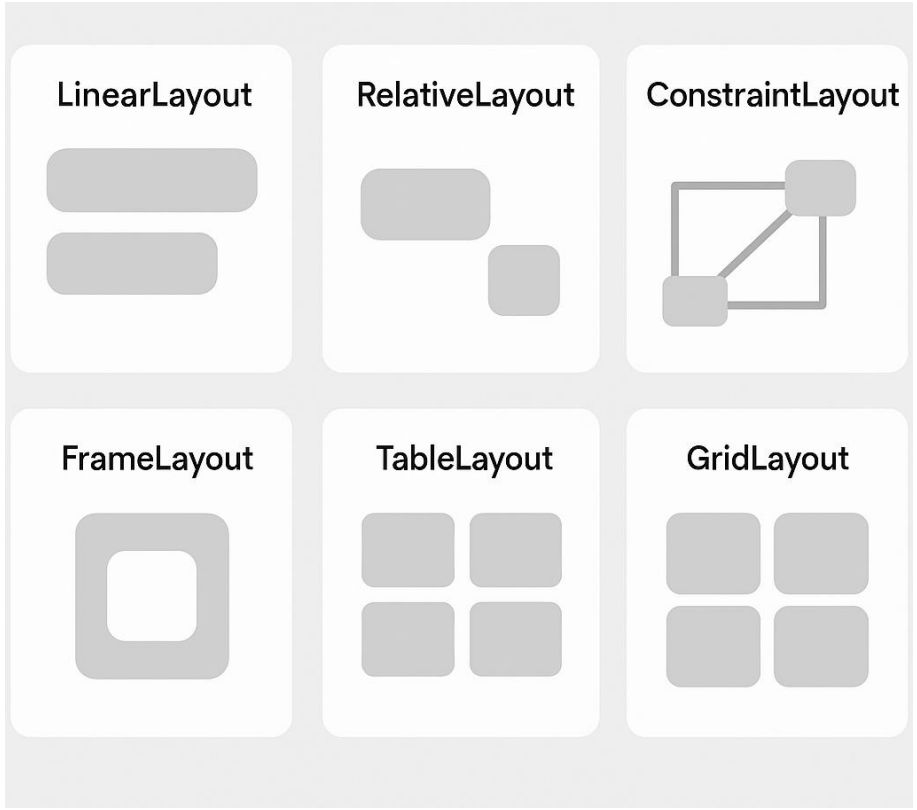
```
<TableLayout>
        <TableRow>
                <TextView android:text="Kolumna 1" />
                <TextView android:text="Kolumna 2" />
        </TableRow>
</TableLayout>
```



# Grid Layout

Podobny do TableLayout, ale bardziej elastyczny — układ siatki z możliwością scalania kolumn/wierszy.

```
<GridLayout
        android:rowCount="2"
        android:columnCount="2">
        <TextView android:text="1" />
        <TextView android:text="2" />
        <TextView android:text="3" />
        <TextView android:text="4" />
</GridLayout>
```



# Button w XML


**Właściwości Button:**

- **Text**: Tekst wyświetlany na przycisku.
  Przykład: Text="Kliknij mnie".
- **TextColor**: Kolor tekstu na przycisku.
  Przykład: TextColor="White".
- **BackgroundColor**: Kolor tła przycisku.
  Przykład: BackgroundColor="Blue".
- **Command**: Akcja lub metoda, która zostanie wywołana po naciśnięciu przycisku (zwykle używana z MVVM).
  Przykład: Command="{Binding ClickCommand}".

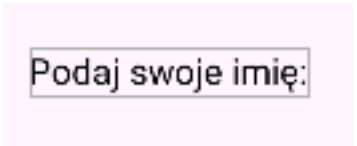
```
<Button
android:id="@+id/myButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Kliknij mnie"
android:textColor="#FFFFFF"
android:background="#2196F3"
android:padding="16dp"
android:layout_marginTop="20dp"
android:layout_gravity="center"/>
```

# Label w XML

**Właściwości Label:**

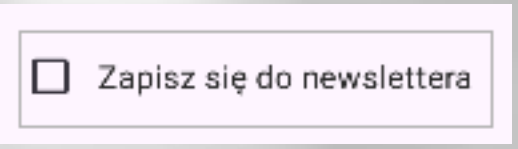
- **Text**: Tekst wyświetlany w etykiecie.
  Przykład: Text="Witaj Świecie".
- **FontSize**: Rozmiar czcionki tekstu.
  Przykład: FontSize="20".
- **TextColor**: Kolor tekstu.
  Przykład: TextColor="Black".
- **HorizontalTextAlignment / VerticalTextAlignment**:
  Wyrównanie tekstu w poziomie i pionie.
  Przykład: HorizontalTextAlignment="Center".

```
<TextView
 android:id="@+id/label1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Podaj swoje imię:"
android:textSize="18sp"
android:textColor="#000000"
android:layout_marginTop="16dp"/>
```



# CheckBox w XML

**Właściwości CheckBox:**




**IsChecked**: Wartość logiczna wskazująca, czy pole wyboru jest zaznaczone.

- Przykład: IsChecked="true".

**Color**: Kolor zaznaczenia w CheckBox.

- Przykład: Color="Green".

**CheckedChanged**: Wydarzenie wywoływane, gdy stan CheckBox zostaje zmieniony.

- Przykład: CheckedChanged="OnCheckBoxChanged".

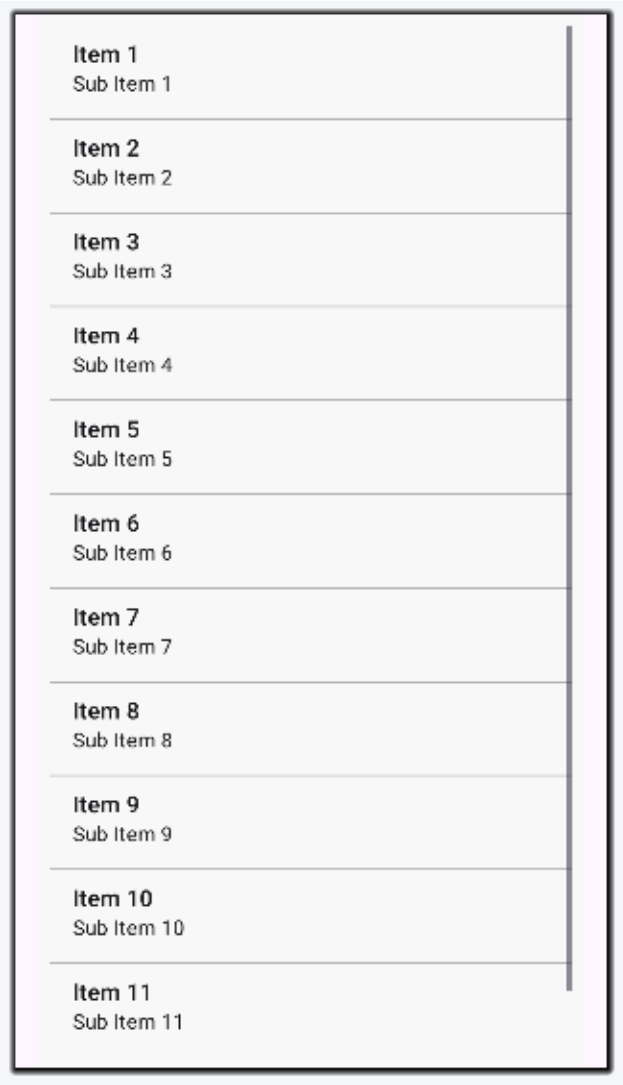
# Przykładowy checkbox

```
<CheckBox
android:id="@+id/checkbox_newsletter"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Zapisz się do newslettera"
android:textSize="16sp"    android:textColor="#333333"
android:padding="8dp"/>
```

**Nie ma odpowiednika GroupBox!**

Można stosować tzw. CardView i na nim osadzić wszystkie kontrolki z grupy.

```
<androidx.cardview.widget.CardView
android:layout_width="match_parent"
android:layout_height="wrap_content"
 android:layout_margin="16dp"
 app:cardCornerRadius="12dp"
app:cardElevation="6dp"
android:background="#FFFFFF">
 <LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
 android:orientation="vertical"
android:padding="16dp">
<TextView
 android:id="@+id/groupbox_title"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
 android:text="Opcje subskrypcji"
android:textSize="18sp"
 android:textStyle="bold"
 android:textColor="#000000"
android:layout_marginBottom="8dp" />
<CheckBox
 android:id="@+id/checkbox_email"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
 android:text="Powiadomienia e-mail"
android:textSize="16sp" />
 <CheckBox
android:id="@+id/checkbox_sms"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Powiadomienia SMS"
 android:textSize="16sp" />
 </LinearLayout>
</androidx.cardview.widget.CardView>
```




# ListView w XML

**Właściwości ListView:**

**ItemsSource**: Kolekcja, która definiuje źródło danych dla ListView.

- Przykład: ItemsSource="{Binding ListaElementow}".

**SelectedItem**: Obiekt aktualnie wybrany w ListView.

- Przykład: SelectedItem="{Binding WybranyElement}".

**SelectionMode**: Tryb wyboru elementów (pojedynczy lub wielokrotny wybór).

- Przykład: SelectionMode="Single".

# Przykładowy ListView

```
<ListView    android:id="@+id/myListView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:divider="#BDBDBD"
android:dividerHeight="1dp"
android:padding="8dp"
android:layout_margin="16dp"
android:background="#FAFAFA" />
```

Objaśnienie:

- android:divider – kolor linii oddzielającej elementy.
- android:dividerHeight – wysokość tej linii.
- android:padding – wewnętrzny margines.
- android:background – kolor tła.
- android:layout_margin – margines wokół listy.

# Ikony w XML

| | |
|---|---|
| **Właściwości Ikon (przy użyciu kontrolki Image):** | |
| **Source**: Określa źródło obrazu (ikon). Może to być lokalny plik, zasób lub URL. | • Przykład: Source="icon.png". |
| **Aspect**: Definiuje sposób wyświetlania ikony w kontrolce. Możliwe wartości to: Fill, AspectFit, AspectFill. | • Przykład: Aspect="AspectFit". |
| **HeightRequest / WidthRequest**: Określa wysokość i szerokość ikony. | • Przykład: HeightRequest="40" WidthRequest="40". |
| **HorizontalOptions / VerticalOptions**: Określa wyrównanie obrazu w układzie. | • Przykład: HorizontalOptions="Center" VerticalOptions="Center". |

# Przykładowa ikona

**Objaśnienie:**
•android:src – ścieżka do zasobu graficznego (może być .png, .jpg, lub vector drawable z res/drawable).
•android:layout_width / height – rozmiar obrazka.
•android:scaleType – sposób skalowania (centerInside, fitCenter, centerCrop itp.).
•android:contentDescription – opis dla dostępności (ważne!).
•android:background – kolor tła za obrazkiem.
•android:padding – odstęp wewnętrzny.
•android:layout_margin – margines na zewnątrz.

```
<ImageView

android:id="@+id/iconImage"
android:layout_width="64dp"
android:layout_height="64dp"
android:src="@drawable/ic_launcher_foreground"

android:contentDescription="Ikona aplikacji"

android:layout_margin="16dp"
android:scaleType="centerInside"
android:background="#E0E0E0"
android:padding="8dp"
android:layout_gravity="center"/>
```

# Menu w XML

**Właściwości Menu (przy użyciu ToolbarItem i MenuItem):**

**ToolbarItem (Górne menu narzędziowe w aplikacji mobilnej):**

**Text**: Tekst wyświetlany na pozycji menu.

- Przykład: Text="Ustawienia".

**IconImageSource**: Określa ikonę dla elementu menu.

- Przykład: IconImageSource="settings_icon.png".

**Order**: Pozycjonowanie elementu w górnym pasku narzędzi (Primary lub Secondary).

- Przykład: Order="Primary".

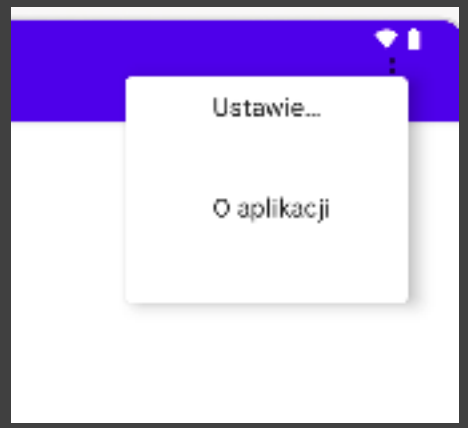
**Priority**: Określa priorytet wyświetlania (niższe wartości mają wyższy priorytet).

- Przykład: Priority="0".

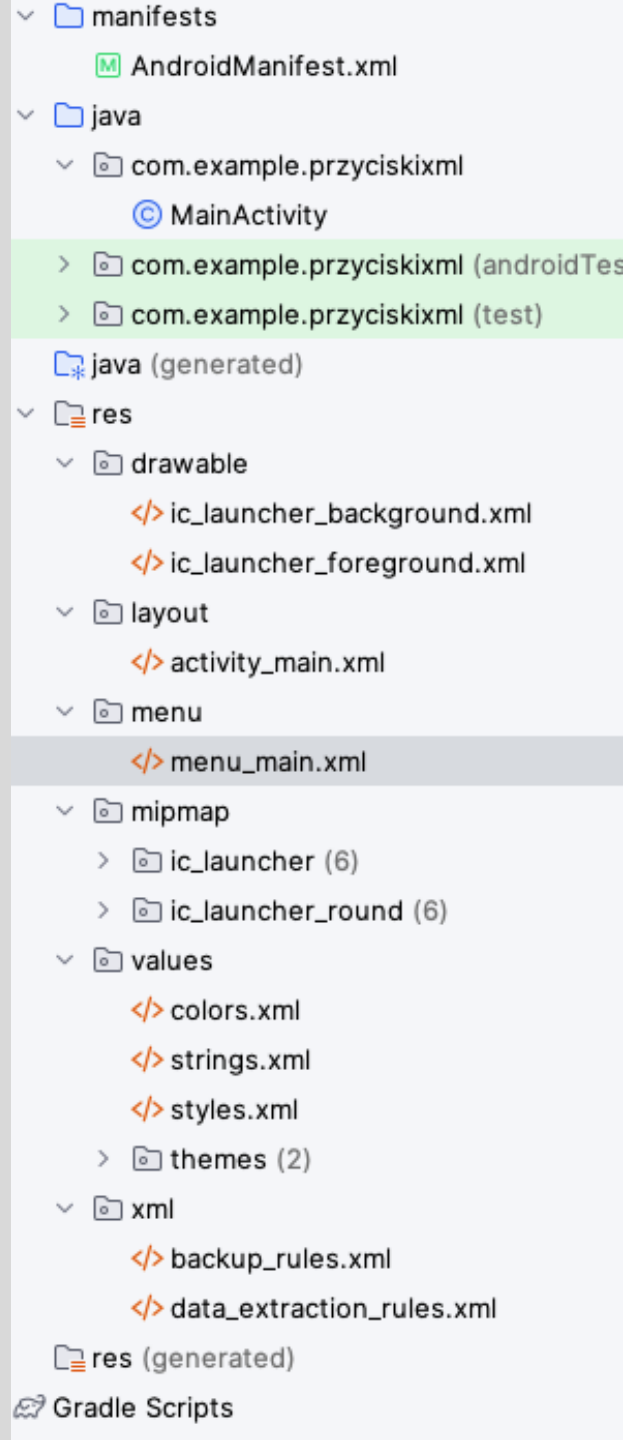
**Command**: Akcja, którą element menu wykonuje po kliknięciu (związana z modelem MVVM).

- Przykład: Command="{Binding SettingsCommand}".

# Przykład ToolbarItem

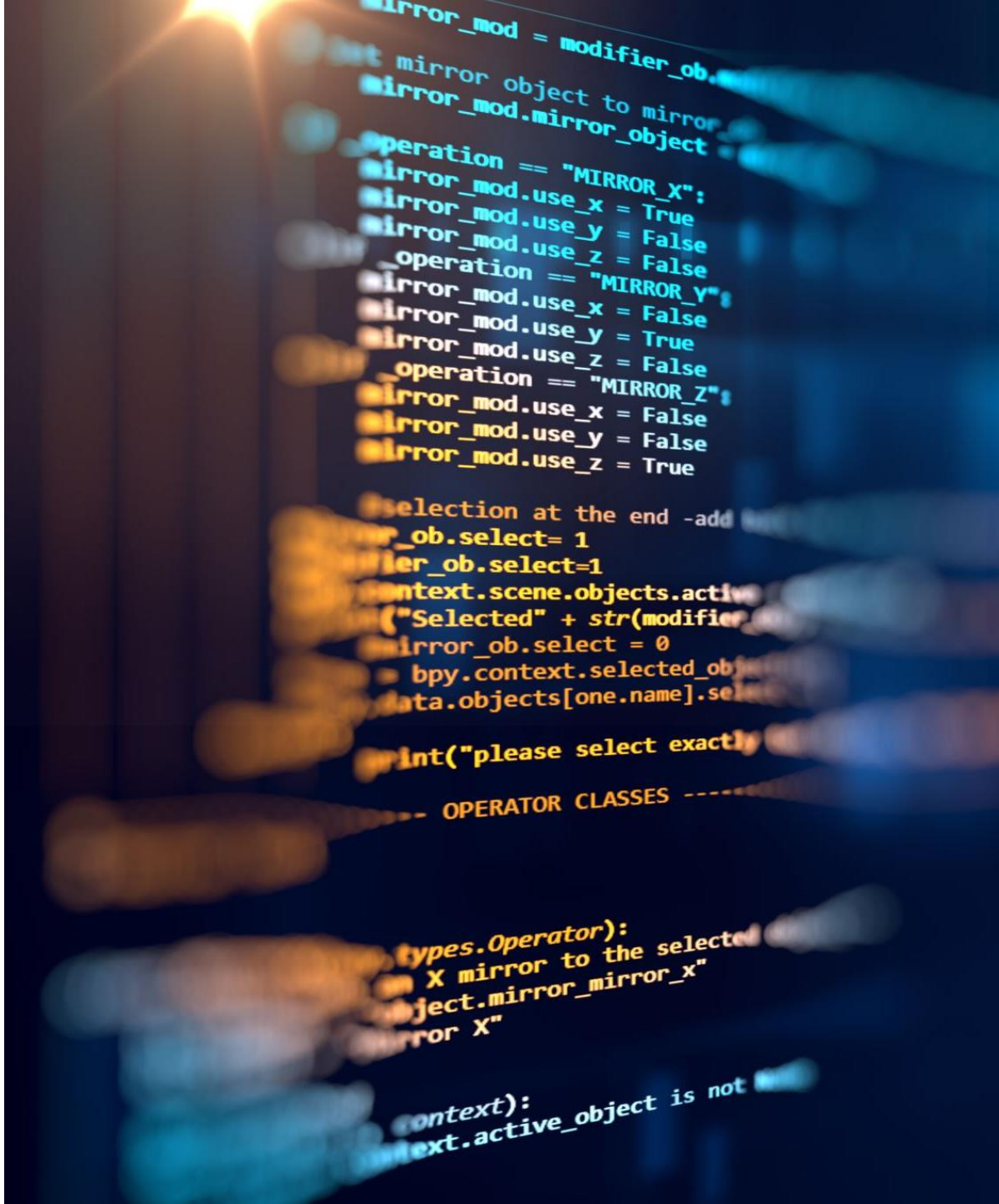



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto">

  <item
    android:id="@+id/action_settings"
    android:title="Ustawienia"
    android:icon="@android:drawable/ic_menu_preferences"
    app:showAsAction="never" />

  <item
    android:id="@+id/action_about"
    android:title="O aplikacji"
    android:icon="@android:drawable/ic_menu_info_details"
    app:showAsAction="never" />

</menu>
```

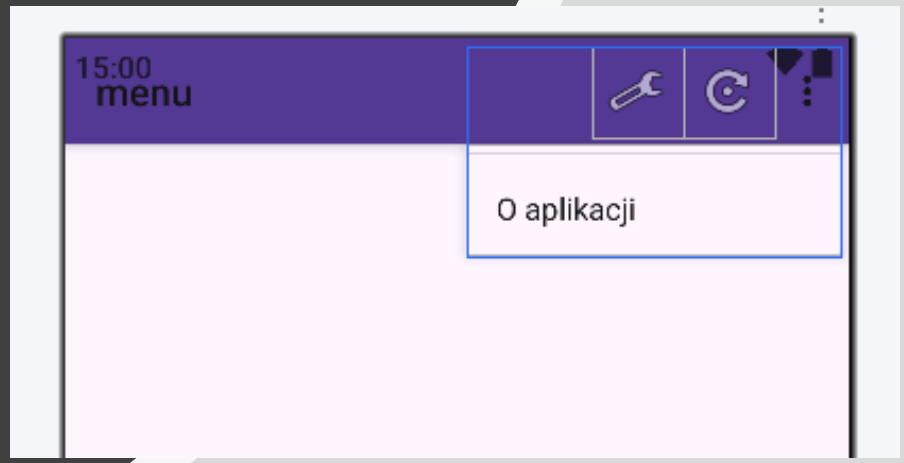


# MenuItem (Element menu w kontekście np. ListView):

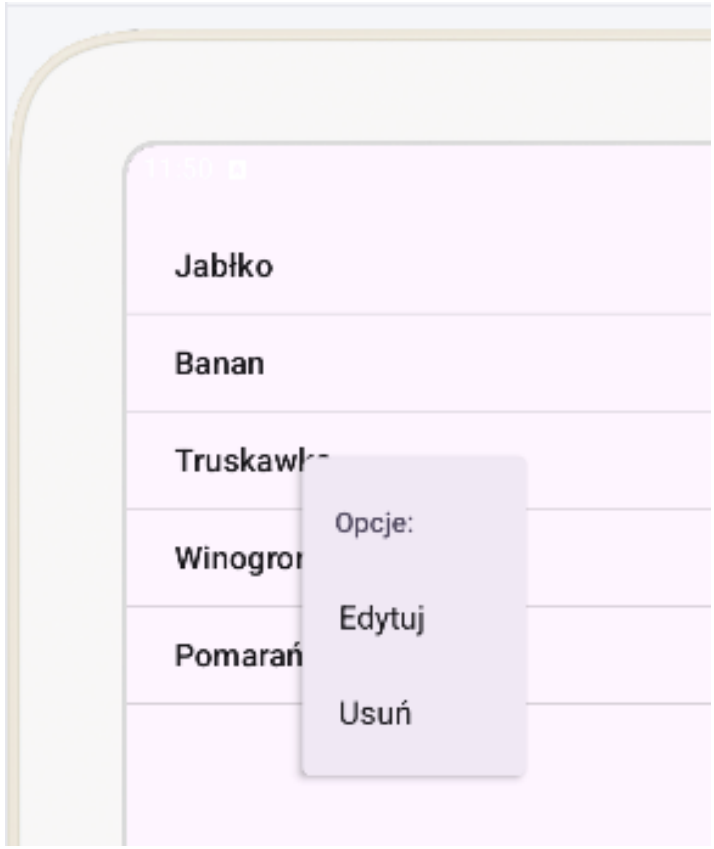
- **Text**: Tekst wyświetlany w elemencie menu.Przykład: Text="Usuń".
- **IconImageSource**: Ikona wyświetlana obok tekstu w menu.Przykład: IconImageSource="delete_icon.png".
- **IsDestructive**: Jeśli ustawione na true, menu zostanie oznaczone jako destrukcyjne (np. usuń).Przykład: IsDestructive="true".
- **Command**: Działanie wykonywane po kliknięciu pozycji menu.
  Przykład: Command="{Binding DeleteCommand}".



# Przykład MenuItem




```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto">

  <!-- Przycisk Ustawienia - jeśli jest miejsce, pokaż na pasku -->
  <item
    android:id="@+id/action_settings"
    android:title="Ustawienia"
    android:icon="@android:drawable/ic_menu_preferences"
    app:showAsAction="ifRoom" />

  <!-- Przycisk O aplikacji - tylko w rozwijanym menu -->
  <item
    android:id="@+id/action_about"
    android:title="O aplikacji"
    android:icon="@android:drawable/ic_menu_info_details"
    app:showAsAction="never" />

  <!-- Przycisk Odśwież - zawsze pokazuj na pasku -->
  <item
    android:id="@+id/action_refresh"
    android:title="Odśwież"
    android:icon="@android:drawable/ic_menu_rotate"
    app:showAsAction="always" />

</menu>
```




# Kontekstowe Menu w ListView

```
<?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/my_listview"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

</LinearLayout>
```

```
package com.example.listaowocw;

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.appcompat.app.AppCompatActivity;

public class MainActivity extends
AppCompatActivity {

  ListView myListView;
  String[] items = {"Jabłko", "Banan", "Truskawka",
"Winogrono", "Pomarańcza"};
  ArrayAdapter<String> adapter;

  int selectedItemIndex = -1;

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

    myListView = findViewById(R.id.my_listview);
    adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, items);
    myListView.setAdapter(adapter);

    // Zarejestruj ListView do menu kontekstowego
    registerForContextMenu(myListView);

    // Obsługa kliknięcia (opcjonalna)
    myListView.setOnItemClickListener((parent,
view, position, id) -> {
      String item = items[position];
      Toast.makeText(this, "Kliknięto: " + item,
Toast.LENGTH_SHORT).show();
    });
  }

  @Override
  public void onCreateContextMenu(ContextMenu
menu, View v, ContextMenu.ContextMenuInfo
menuInfo) {
    super.onCreateContextMenu(menu, v,
menuInfo);

    AdapterView.AdapterContextMenuInfo info =
(AdapterView.AdapterContextMenuInfo) menuInfo;
    selectedItemIndex = info.position;

    menu.setHeaderTitle("Opcje:");
    menu.add(0, v.getId(), 0, "Edytuj");
    menu.add(0, v.getId(), 1, "Usuń");
  }

  @Override
  public boolean onContextItemSelected(MenuItem
item) {
    if (item.getTitle() == "Edytuj") {
      Toast.makeText(this, "Edytuj: " +
items[selectedItemIndex],
Toast.LENGTH_SHORT).show();
    } else if (item.getTitle() == "Usuń") {
      Toast.makeText(this, "Usunięto: " +
items[selectedItemIndex],
Toast.LENGTH_SHORT).show();
    } else {
      return false;
    }
    return true;
  }
}
```

# Przykładowa prosta aplikacja z zastosowaniem różnych widgetów

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/and
roid"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

<LinearLayout
  android:orientation="vertical"
  android:padding="16dp"
  android:layout_width="match_parent"
  android:layout_height="wrap_content">

  <!-- CUSTOM TOOLBAR -->
  <androidx.appcompat.widget.Toolbar
    android:id="@+id/customToolbar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="?attr/colorPrimary">

    <TextView
      android:id="@+id/toolbar_title"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center"
      android:text="Moja Aplikacja"
      android:textColor="@android:color/white"
      android:textSize="20sp"
      android:textStyle="bold"
      android:padding="12dp"/>
  </androidx.appcompat.widget.Toolbar>

  <!-- RESZTA WIDGETÓW -->

  <EditText
    android:id="@+id/nameInput"
    android:hint="Wpisz swoje imię"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <RadioGroup
    android:id="@+id/genderGroup"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <RadioButton android:id="@+id/radioMale"
android:text="Mężczyzna"/>
    <RadioButton android:id="@+id/radioFemale"
android:text="Kobieta"/>
  </RadioGroup>

  <CheckBox
    android:id="@+id/checkboxTerms"
    android:text="Akceptuję regulamin"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

  <Switch
    android:id="@+id/darkModeSwitch"
    android:text="Tryb ciemny"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

  <SeekBar
    android:id="@+id/volumeSeek"
    android:max="100"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <Spinner
    android:id="@+id/spinnerCity"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <Button
    android:id="@+id/showBtn"
    android:text="Pokaż dane"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <ProgressBar
    android:id="@+id/progressBar"
    android:visibility="gone"
    style="?android:attr/progressBarStyleHorizontal"
    android:max="100"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <ImageView
    android:id="@+id/imageView"
    android:src="@drawable/sample_image"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:scaleType="centerCrop"/>

  <ListView
    android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="200dp"/>

</LinearLayout>
</ScrollView>
```

**Przykładowa aplikacja z zastosowaniem różnych widgetów**

```
package com.example.mywidgetsapp;


import android.os.Bundle;
import android.view.View;
import android.widget.*;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;


public class MainActivity extends AppCompatActivity {


  EditText nameInput;
  RadioGroup genderGroup;
  CheckBox checkboxTerms;
  Switch darkModeSwitch;
  SeekBar volumeSeek;
  Spinner spinnerCity;
  Button showBtn;
  ProgressBar progressBar;
  ImageView imageView;
  ListView listView;


  String[] cities = {"Warszawa", "Kraków", "Gdańsk", "Wrocław"};


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


    // Ustawienie customowego Toolbaru
    Toolbar toolbar = findViewById(R.id.customToolbar);
    setSupportActionBar(toolbar);
    if (getSupportActionBar() != null) {
      getSupportActionBar().setDisplayShowTitleEnabled(false); // Ukryj domyślny tytuł
    }


    TextView title = findViewById(R.id.toolbar_title);
    title.setText("Moja Aplikacja");


    nameInput = findViewById(R.id.nameInput);
    genderGroup = findViewById(R.id.genderGroup);
    checkboxTerms = findViewById(R.id.checkboxTerms);
    darkModeSwitch = findViewById(R.id.darkModeSwitch);
    volumeSeek = findViewById(R.id.volumeSeek);
    spinnerCity = findViewById(R.id.spinnerCity);
    showBtn = findViewById(R.id.showBtn);
    progressBar = findViewById(R.id.progressBar);
    imageView = findViewById(R.id.imageView);
    listView = findViewById(R.id.listView);


    ArrayAdapter<String> cityAdapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item, cities);
    spinnerCity.setAdapter(cityAdapter);


    showBtn.setOnClickListener(v -> {
      progressBar.setVisibility(View.VISIBLE);
      progressBar.setProgress(50);


      String name = nameInput.getText().toString();
      String gender = "";
      int selectedId = genderGroup.getCheckedRadioButtonId();
      if (selectedId != -1) {
        RadioButton selectedGender = findViewById(selectedId);
        gender = selectedGender.getText().toString();
      }


      boolean termsAccepted = checkboxTerms.isChecked();
      boolean isDarkMode = darkModeSwitch.isChecked();
      int volume = volumeSeek.getProgress();
      String selectedCity = spinnerCity.getSelectedItem().toString();


      String[] summary = {
        "Imię: " + name,
        "Płeć: " + gender,
        "Miasto: " + selectedCity,
        "Ciemny tryb: " + (isDarkMode ? "Tak" : "Nie"),
        "Regulamin: " + (termsAccepted ? "Zaakceptowany" : "Niezaakceptowany"),
        "Głośność: " + volume + "%"
      };


      ArrayAdapter<String> listAdapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, summary);
      listView.setAdapter(listAdapter);
      progressBar.setProgress(100);
    });
  }
}
```

# Częsty błąd

```
<resources xmlns:tools="http://schemas.android.com/tools">
   <!-- Base application theme. -->
   <style name="Theme.Widgety"
parent="Theme.MaterialComponents.DayNight.NoActionBar">
      <!-- Primary brand color. -->
      <item name="colorPrimary">@color/purple_500</item>
      <item
name="colorPrimaryVariant">@color/purple_700</item>
      <item name="colorOnPrimary">@color/white</item>
      <!-- Secondary brand color. -->
      <item name="colorSecondary">@color/teal_200</item>
      <item
name="colorSecondaryVariant">@color/teal_700</item>
      <item name="colorOnSecondary">@color/black</item>
      <!-- Status bar color. -->
      <item
name="android:statusBarColor">?attr/colorPrimaryVariant</item>
      <!-- Customize your theme here. -->
   </style>
</resources>
```

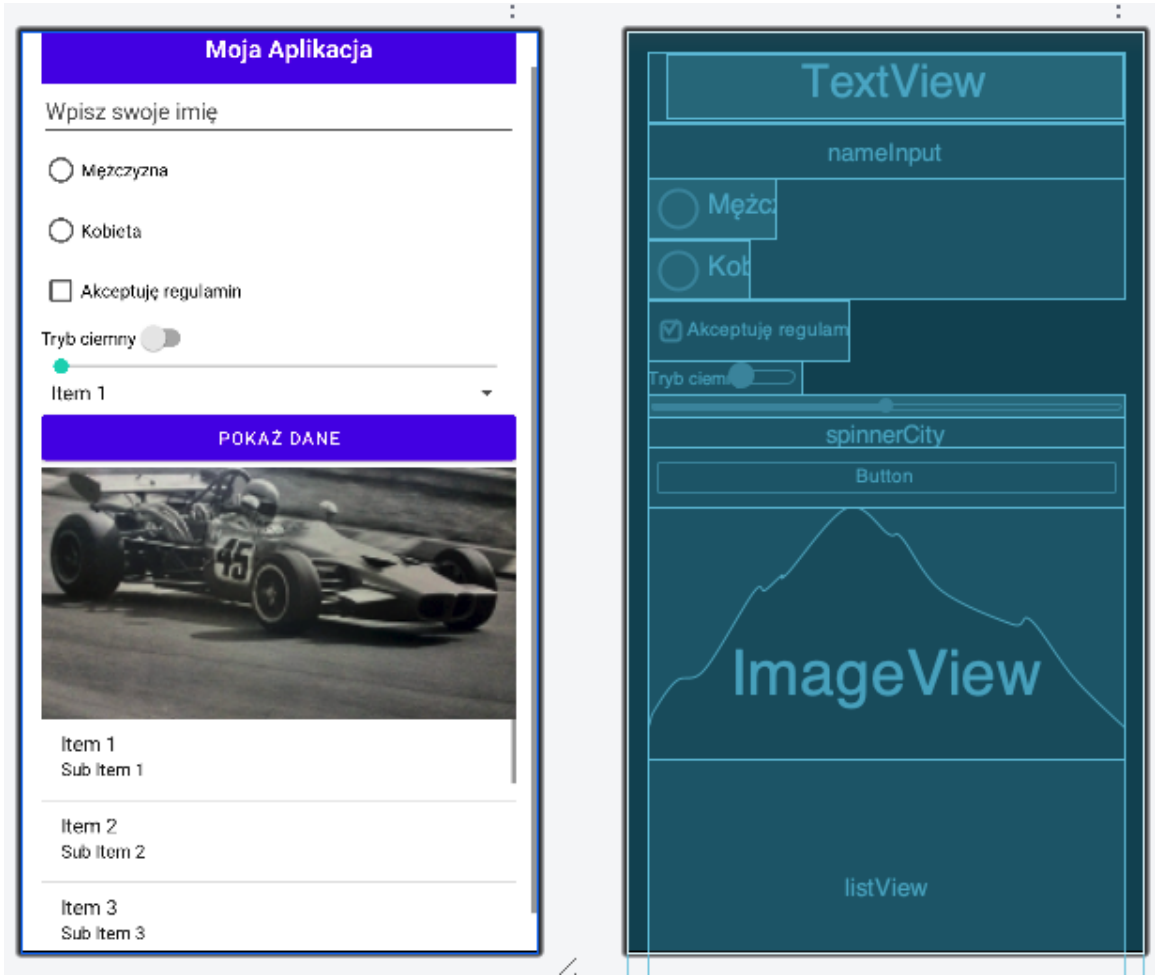
Plik themes.xml często zawiera pułapki, które powodują problemy w działaniu kodu.

Dla Android Studio błędem może być nawet zły dobór kolorów. W wielu przypadkach problemem będzie umieszczanie elementów warstwami.

# Efekt działania programu

Widok z edytora graficznego:

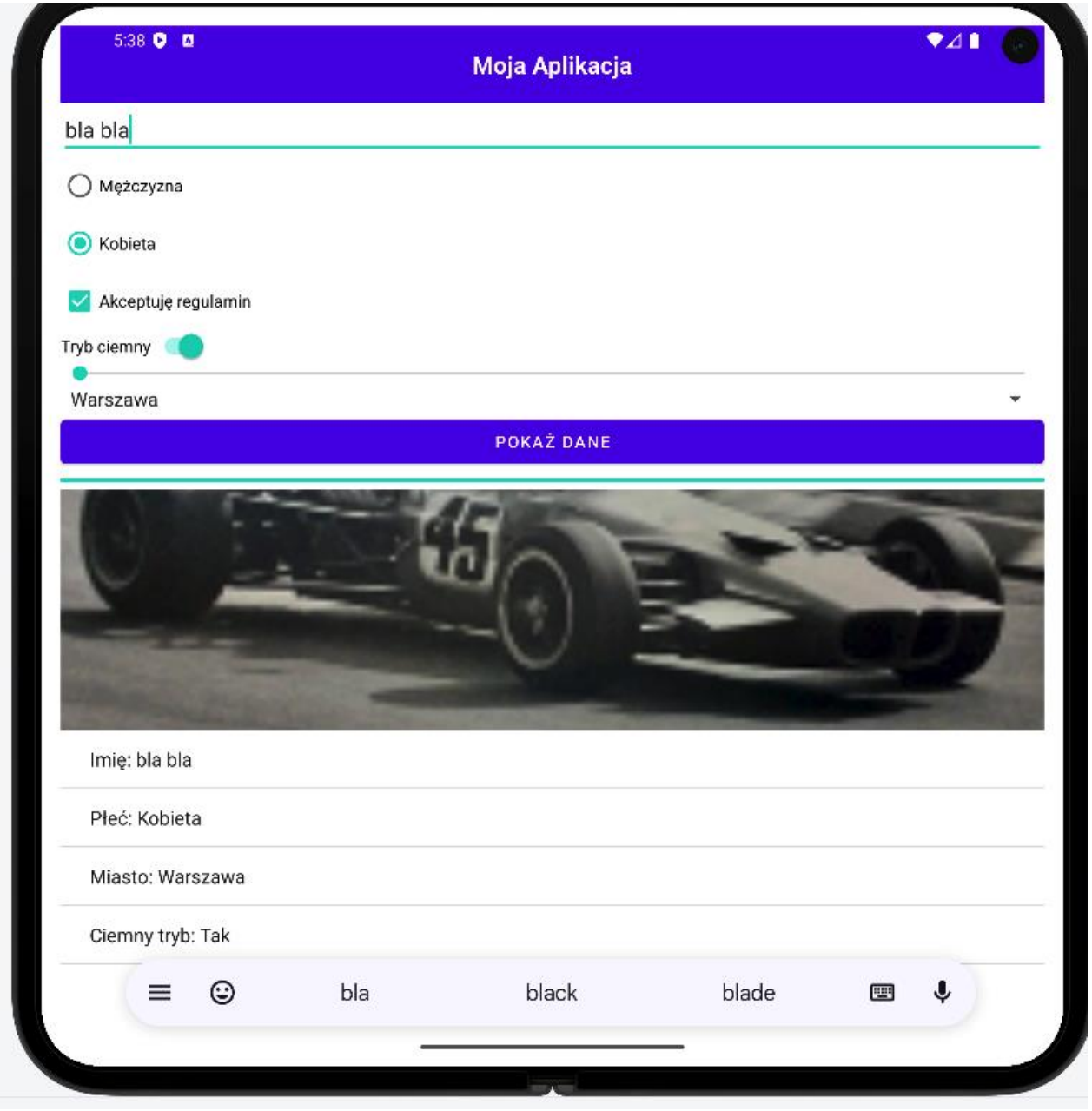
Aplikacja w formie formularza pozwala wyświetlać dane pobrane od użytkownika i po naciśnięciu przycisku wyświetla dane wpisane w pola edycyjne lub zwraca wybór z zaznaczonych pól.



# Efekt działania programu

## Widok z emulatora

**Aplikacja dzięki właściwościom WRAP_CONTENT I MATCH_PARENT dostosuje się do rozmiarów każdego ekranu.**



# Typowa aplikacja – konwerter jednostek


```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.przelicznik">

  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="Przelicznik"

android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Przelicznik">

    <activity
      android:name=".MainActivity"
      android:exported="true"> <!-- <<< DODANE TUTAJ -->
      <intent-filter>
        <action
android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCHER"
/>
      </intent-filter>
    </activity>
  </application>

</manifest>
```

Manifest

# Plik JAVA

```
ackage com.example.przelicznik;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private EditText editValue;
    private TextView tvResult;

    private Button btnFromMeters, btnFromCentimeters, btnFromInches;
    private Button btnToMeters, btnToCentimeters, btnToInches;
    private Button btnConvert;

    // Zmienna do przechowywania wybranej jednostki źródłowej i docelowej
    private String fromUnit = "";
    private String toUnit = "";

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

        // Znajdź widoki
        editValue = findViewById(R.id.editValue);
        tvResult = findViewById(R.id.tvResult);

        btnFromMeters = findViewById(R.id.btnFromMeters);
        btnFromCentimeters = findViewById(R.id.btnFromCentimeters);
        btnFromInches = findViewById(R.id.btnFromInches);

        btnToMeters = findViewById(R.id.btnToMeters);
        btnToCentimeters = findViewById(R.id.btnToCentimeters);
        btnToInches = findViewById(R.id.btnToInches);

        btnConvert = findViewById(R.id.btnConvert);

        // Ustaw kliknięcia przycisków wyboru jednostek źródłowych
        btnFromMeters.setOnClickListener(v -> selectFromUnit("m"));
        btnFromCentimeters.setOnClickListener(v -> selectFromUnit("cm"));
        btnFromInches.setOnClickListener(v -> selectFromUnit("in"));

        // Ustaw kliknięcia przycisków wyboru jednostek docelowych
        btnToMeters.setOnClickListener(v -> selectToUnit("m"));
        btnToCentimeters.setOnClickListener(v -> selectToUnit("cm"));
        btnToInches.setOnClickListener(v -> selectToUnit("in"));

        // Kliknięcie przycisku Przelicz
        btnConvert.setOnClickListener(v -> convertValue());
    }

    private void selectFromUnit(String unit) {
        fromUnit = unit;
        // Zmień kolor wybranego przycisku (źródłowego), resetuj inne
        resetFromButtons();
        switch (unit) {
            case "m":
btnFromMeters.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark));
break;
            case "cm":
btnFromCentimeters.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark));
break;
            case "in":
btnFromInches.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark)); break;
        }
    }

    private void resetFromButtons() {
        btnFromMeters.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));

btnFromCentimeters.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));
        btnFromInches.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));
    }

    private void selectToUnit(String unit) {
        toUnit = unit;
        // Zmień kolor wybranego przycisku (docelowego), resetuj inne
        resetToButtons();
        switch (unit) {
            case "m":
btnToMeters.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark)); break;
            case "cm":
btnToCentimeters.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark));
break;
            case "in":
btnToInches.setBackgroundTintList(getColorStateList(android.R.color.holo_blue_dark)); break;
        }
    }

    private void resetToButtons() {
        btnToMeters.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));
        btnToCentimeters.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));
        btnToInches.setBackgroundTintList(getColorStateList(android.R.color.holo_red_dark));
    }

    private void convertValue() {
        String input = editValue.getText().toString().trim();

        if (input.isEmpty()) {
            Toast.makeText(this, "Wpisz wartość do przeliczenia", Toast.LENGTH_SHORT).show();
            return;
        }

        if (fromUnit.isEmpty()) {
            Toast.makeText(this, "Wybierz jednostkę źródłową", Toast.LENGTH_SHORT).show();
            return;
        }

        if (toUnit.isEmpty()) {
            Toast.makeText(this, "Wybierz jednostkę docelową", Toast.LENGTH_SHORT).show();
            return;
        }

        double value;
        try {
            value = Double.parseDouble(input);
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Niepoprawna liczba", Toast.LENGTH_SHORT).show();
            return;
        }

        // Konwersja wartości do metrów
        double valueInMeters = convertToMeters(value, fromUnit);

        // Konwersja z metrów do wybranej jednostki docelowej
        double result = convertFromMeters(valueInMeters, toUnit);

        // Wyświetl wynik z 4 miejscami po przecinku
        tvResult.setText(String.format("%.4f %s", result, getUnitFullName(toUnit)));
    }

    private double convertToMeters(double val, String unit) {
        switch (unit) {
            case "m": return val;
            case "cm": return val / 100.0;
            case "in": return val * 0.0254;
            default: return val;
        }
    }

    private double convertFromMeters(double val, String unit) {
        switch (unit) {
            case "m": return val;
            case "cm": return val * 100.0;
            case "in": return val / 0.0254;
            default: return val;
        }
    }

    private String getUnitFullName(String unit) {
        switch (unit) {
            case "m": return "metrów";
            case "cm": return "centymetrów";
            case "in": return "cal(i)";
            default: return "";
        }
    }
}
```

# Plik XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="24dp"
    tools:context=".MainActivity">

    <!-- Niestandardowy pasek tytułu z ikoną JPG -->
    <LinearLayout
        android:id="@+id/custom_toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="?attr/colorPrimary"
        android:gravity="center_vertical"
        android:orientation="horizontal"
        android:paddingStart="16dp"
        android:paddingEnd="16dp">

        <ImageView
            android:id="@+id/toolbar_icon"
            android:layout_width="32dp"
            android:layout_height="32dp"
            android:layout_marginEnd="8dp"
            android:contentDescription="Ikona"
            android:src="@drawable/ic_baseline_swap_horiz_24"
            android:scaleType="centerCrop" />

        <TextView
            android:id="@+id/toolbar_title"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Przelicznik"
            android:textColor="#FFFFFF"
            android:textSize="20sp"
            android:gravity="center"
            android:textStyle="bold"/>
    </LinearLayout>

    <!-- Pole do wpisywania wartości -->
    <EditText
        android:id="@+id/editValue"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:inputType="numberDecimal"
        android:hint="Wpisz wartość"
        android:textSize="24sp"
        android:padding="12dp"
        android:minHeight="60dp"
        android:layout_marginTop="24dp"
        android:gravity="center"/>

    <!-- Jednostka źródłowa -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Wybierz jednostkę źródłową:"
        android:layout_marginTop="16dp"
        android:layout_gravity="center"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginTop="8dp"
        android:gravity="center">

        <Button
            android:id="@+id/btnFromMeters"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Metry"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />

        <Button
            android:id="@+id/btnFromCentimeters"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Centymetry"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />

        <Button
            android:id="@+id/btnFromInches"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Cale"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />
    </LinearLayout>

    <!-- Jednostka docelowa -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Wybierz jednostkę docelową:"
        android:layout_marginTop="24dp"
        android:layout_gravity="center"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginTop="8dp"
        android:gravity="center">

        <Button
            android:id="@+id/btnToMeters"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Metry"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />

        <Button
            android:id="@+id/btnToCentimeters"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Centymetry"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />

        <Button
            android:id="@+id/btnToInches"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:text="Cale"
            android:layout_height="wrap_content"
            android:backgroundTint="#FF0000"
            android:textColor="#FFFFFF" />
    </LinearLayout>

    <!-- Przycisk Przelicz -->
    <Button
        android:id="@+id/btnConvert"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Przelicz"
        android:layout_marginTop="32dp"
        android:backgroundTint="#FF0000"
        android:textColor="#FFFFFF" />

    <!-- Wynik -->
    <TextView
        android:id="@+id/tvResult"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20sp"
        android:textColor="#000000"
        android:gravity="center"
        android:layout_marginTop="32dp" />
</LinearLayout>
```

# Material UI

Elementy i motywy:

- **Kolory i motywy** — definiowane globalnie
- **Typografia** — Roboto jako podstawowy font
- **Komponenty UI**:
- Przyciski (MaterialButton)
- Karty (CardView)
- Paski narzędzi (Toolbar)
- Dialogi, Snackbar, FloatingActionButton
- Formularze: TextInputLayout, TextInputEditText.

**Przykład użycia:**

**Dodaj do build.gradle:**

**implementation 'com.google.android.material:material:1.9.0’**

**Ustaw motyw w res:**

**<style name="Theme.MyApp" parent="Theme.MaterialComponents.DayNight.DarkActionBar">**

**<!-- Kolory i styl -->**

**</style>**

**Sprawdź w manifeście jakiego motywu domyślnie używasz i zmień.**

# Przykład

```
<com.google.android.material.button.MaterialButton
    android:id="@+id/myButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Kliknij mnie"
    style="@style/Widget.MaterialComponents.Button"
    app:cornerRadius="8dp"
    app:icon="@drawable/ic_star"
    app:iconGravity="textStart"/>
```

```
<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Email">

    <com.google.android.material.textfield.TextInputEditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textEmailAddress"/>
</com.google.android.material.textfield.TextInputLayout>
```

# Przykładowy layout z użyciem MaterialUI

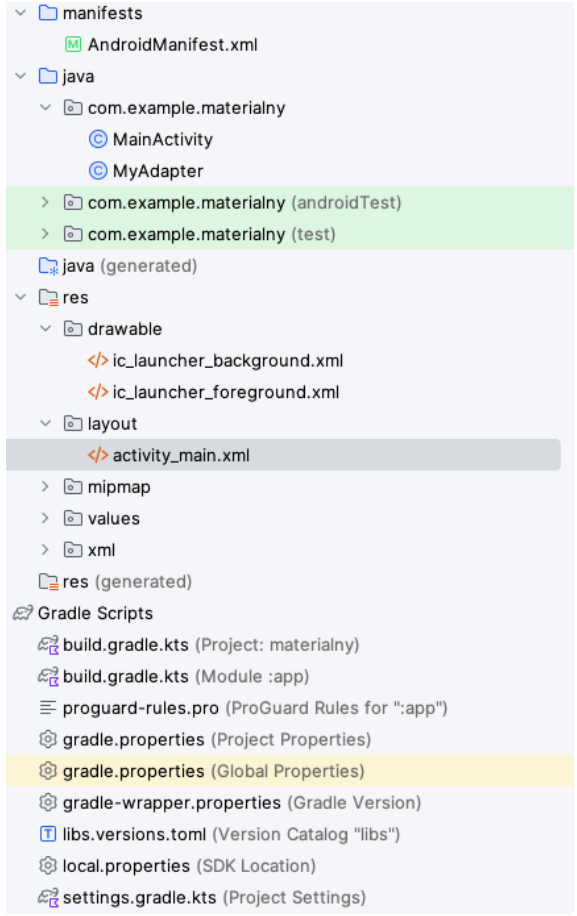

```
<androidx.coordinatorlayout.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity">
  <com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:layout_gravity="center_vertical">
    <com.google.android.material.textfield.TextInputEditText
      android:id="@+id/editText"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Wpisz coś" />
  </com.google.android.material.textfield.TextInputLayout>
  <com.google.android.material.button.MaterialButton
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Kliknij mnie"
    app:layout_anchor="@id/editText"
    app:layout_anchorGravity="bottom|end"
    android:layout_margin="16dp" />
  <androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="150dp" />
</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

# Przykład aplikacji z MaterialUI

```
package com.example.materialny;


import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import com.google.android.material.textfield.TextInputEditText;
import java.util.ArrayList;


public class MainActivity extends AppCompatActivity{


  private TextInputEditText editText;
  private RecyclerView recyclerView;
  private MyAdapter adapter;
  private ArrayList<String> items;


  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    editText = findViewById(R.id.editText);
    Button button = findViewById(R.id.button);
    recyclerView = findViewById(R.id.recyclerView);


    items = new ArrayList<>();
    adapter = new MyAdapter(items);
    recyclerView.setLayoutManager(new LinearLayoutManager(this));
    recyclerView.setAdapter(adapter);


    button.setOnClickListener(new View.OnClickListener(){
      @Override
      public void onClick(View v) {
        String text = editText.getText().toString();
        if (!text.isEmpty()) {
          items.add(text);
          adapter.notifyItemInserted(items.size() - 1);
          editText.setText("");
        }
      }
    });
  }
}
```

```
package com.example.materialny;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import
androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;

public class MyAdapter extends
RecyclerView.Adapter<MyAdapter.ViewHol
der> {

  private ArrayList<String> items;

  public MyAdapter(ArrayList<String> items)
{
    this.items = items;
  }

  @NonNull
  @Override
  public ViewHolder
onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
    View view =
LayoutInflater.from(parent.getContext()).infl
ate(android.R.layout.simple_list_item_1,
parent, false);
    return new ViewHolder(view);
  }

  @Override
  public void onBindViewHolder(@NonNull
ViewHolder holder, int position) {

holder.textView.setText(items.get(position));
  }

  @Override
  public int getItemCount() {
    return items.size();
  }

  public static class ViewHolder extends
RecyclerView.ViewHolder {
    TextView textView;

    public ViewHolder(@NonNull View
itemView) {
      super(itemView);
      textView =
itemView.findViewById(android.R.id.text1);
    }
  }
}
```

# Przykład aplikacji z MaterialUI

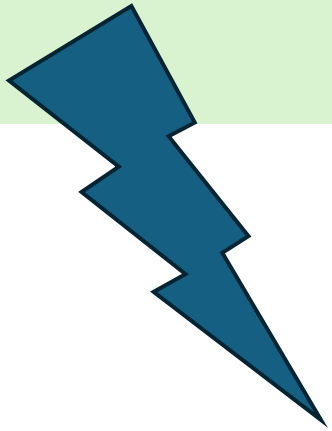



**Co zrobiliśmy?**

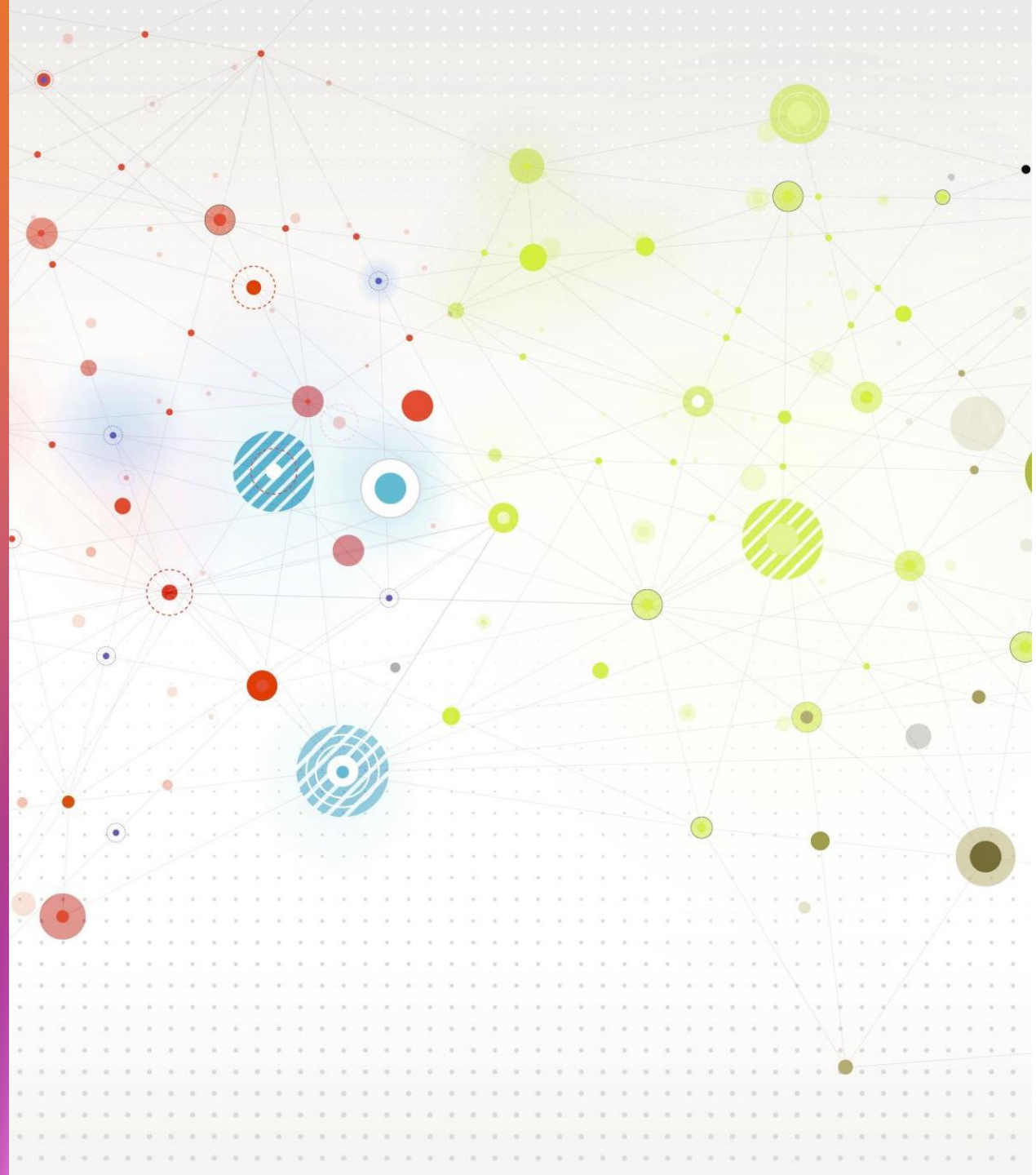
**Dodaliśmy zależność do pliku build.gradle**

**Napisaliśmy XML i Javę**

**Zrobiliśmy Adapter (o tym później, potrzebne gdy przesyłamy informacje).**



**Sprawny adapter to taki, który jest umieszczony na tym samym poziomie co klasy Activity a nie o poziom niżej.**



# Zalety?

- Spójny wygląd aplikacji na Android i web
- Gotowe, przetestowane komponenty
- Responsywność i animacje
- Łatwa personalizacja i rozbudowa
- Wsparcie przez Google i dużą społeczność

# Uprawnienia w Androidzie

**Lokalizacja:**
•ACCESS_FINE_LOCATION – dokładna lokalizacja (GPS)
•ACCESS_COARSE_LOCATION – przybliżona lokalizacja (Wi-Fi, BTS)
•ACCESS_BACKGROUND_LOCATION – lokalizacja w tle (Android 10+).

**Pamięć urządzenia:**
• READ_EXTERNAL_STORAGE – odczyt danych z pamięci
• WRITE_EXTERNAL_STORAGE – zapis do pamięci
⚠ Od Androida 10+ należy stosować Scoped Storage
• MANAGE_EXTERNAL_STORAGE – pełen dostęp do pamięci (Android 11+ – tylko dla aplikacji specjalnych)

**Aparat i multimedia:**
• CAMERA – dostęp do aparatu
• RECORD_AUDIO – nagrywanie dźwięku
• READ_MEDIA_IMAGES – odczyt zdjęć (Android 13+)
• READ_MEDIA_VIDEO – odczyt wideo (Android 13+)
• READ_MEDIA_AUDIO – odczyt plików audio (Android 13+)

# Uprawnienia w Androidzie

**Telefon:**
- CALL_PHONE – wykonywanie połączeń
- READ_PHONE_STATE – odczyt statusu połączeń i IMEI
- ANSWER_PHONE_CALLS – odbieranie połączeń (Android 8+)
- READ_CALL_LOG – odczyt historii połączeń
- WRITE_CALL_LOG – edycja historii połączeń
- ADD_VOICEMAIL – dodawanie poczty głosowej
- USE_SIP – korzystanie z połączeń SIP (VoIP)

**Kontakty:**
- READ_CONTACTS – odczyt kontaktów
- WRITE_CONTACTS – modyfikacja kontaktów
- GET_ACCOUNTS – odczyt kont użytkownika (np. Google)

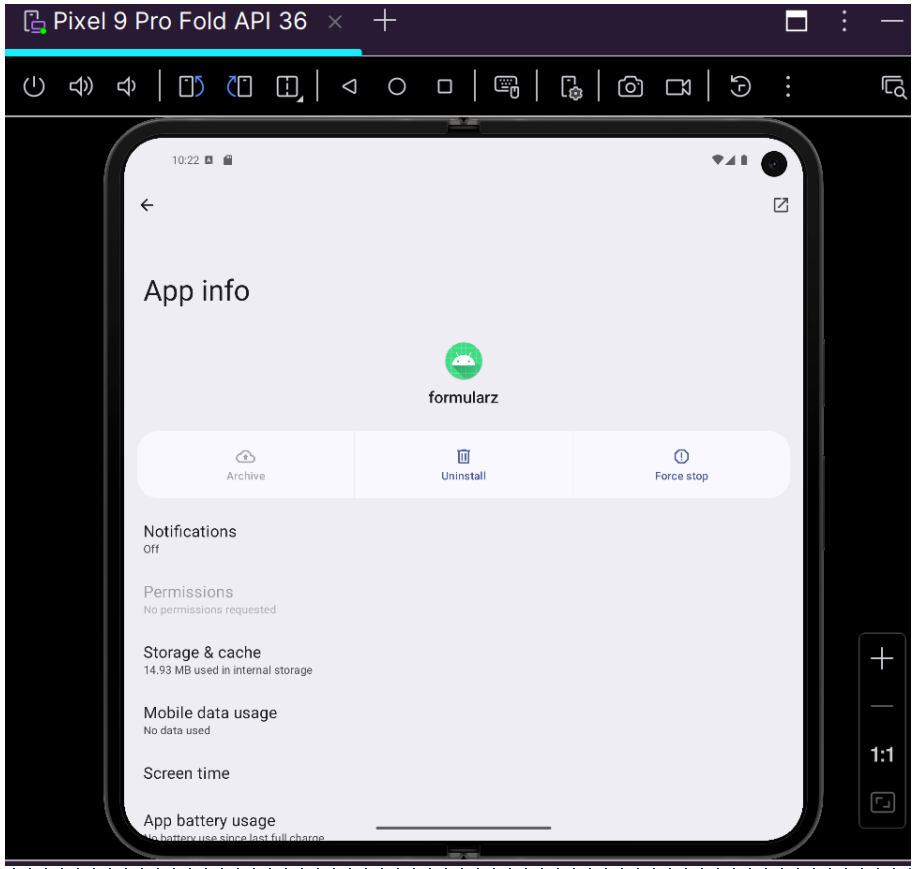
**SMS i wiadomości:**
- SEND_SMS – wysyłanie SMS-ów
- RECEIVE_SMS – odbiór SMS-ów
- READ_SMS – odczyt SMS-ów
- RECEIVE_MMS – odbiór MMS-ów
- RECEIVE_WAP_PUSH – odbiór wiadomości WAP PUSH

# Uprawnienia w Androidzie

READ_CALENDAR – odczyt wydarzeń z kalendarza
WRITE_CALENDAR – dodawanie/edytowanie wydarzeń

BODY_SENSORS – dostęp do danych z czujników ciała (np. tętno)

INTERNET – dostęp do internetu (normalne uprawnienie – nie wymaga zgody)
BLUETOOTH / BLUETOOTH_ADMIN – sterowanie Bluetooth
FOREGROUND_SERVICE – uruchamianie usług w tle (Android 9+)



Plik języka JAVA:

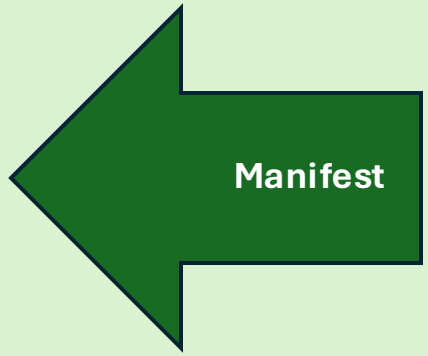
# Przykład aplikacji z uprawnieniami

```
package com.example.uprawnienia;

import android.Manifest;
Import android.app.Activity;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import
com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
public class MainActivity extends Activity {
  private static final int PERMISSION_REQUEST_CODE = 123;
  private TextView infoText;
  private FusedLocationProviderClient fusedLocationClient;
  String[] permissions = {
      Manifest.permission.ACCESS_FINE_LOCATION,
      Manifest.permission.CAMERA,
      Manifest.permission.READ_EXTERNAL_STORAGE
  };
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    infoText = findViewById(R.id.infoText);
    Button locationBtn = findViewById(R.id.btnLocation);
    Button cameraBtn = findViewById(R.id.btnCamera);
    Button storageBtn = findViewById(R.id.btnStorage);
    fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this);
    if (!hasAllPermissions()) {
      ActivityCompat.requestPermissions(this, permissions,
PERMISSION_REQUEST_CODE);
    }
    locationBtn.setOnClickListener(v -> showLocation());
    cameraBtn.setOnClickListener(v -> openCamera());
    storageBtn.setOnClickListener(v -> openGallery());
  }
  private boolean hasAllPermissions() {
    for (String permission : permissions) {
      if (ContextCompat.checkSelfPermission(this, permission) !=
PackageManager.PERMISSION_GRANTED)
        return false;
    }
    return true;
  }
  private void showLocation() {
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
      fusedLocationClient.getLastLocation()
          .addOnSuccessListener(this, location -> {
            if (location != null) {
              infoText.setText("Lat: " + location.getLatitude() + ",
Lon: " + location.getLongitude());
            } else {
              infoText.setText("Brak lokalizacji.");
            }
          });
    } else {
      Toast.makeText(this, "Brak dostępu do lokalizacji",
Toast.LENGTH_SHORT).show();
    }
  }
  private void openCamera() {
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.CAMERA) ==
PackageManager.PERMISSION_GRANTED) {
      Intent cameraIntent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
      startActivity(cameraIntent);
    } else {
      Toast.makeText(this, "Brak dostępu do aparatu",
Toast.LENGTH_SHORT).show();
    }
  }
  private void openGallery() {
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_EXTERNAL_STORAGE) ==
PackageManager.PERMISSION_GRANTED) {
      Intent intent = new Intent(Intent.ACTION_PICK);
      intent.setType("image/*");
      startActivity(intent);
    } else {
      Toast.makeText(this, "Brak dostępu do pamięci",
Toast.LENGTH_SHORT).show();
    }
  }
  @Override
  public void onRequestPermissionsResult(int requestCode,
@NonNull String[] permissions,
                         @NonNull int[] grantResults) {
    if (requestCode == PERMISSION_REQUEST_CODE) {
      if (!hasAllPermissions()) {
        Toast.makeText(this, "Nie udzielono wszystkich
uprawnień", Toast.LENGTH_SHORT).show();
      }
    }
    super.onRequestPermissionsResult(requestCode,
permissions, grantResults);
  }
}
```

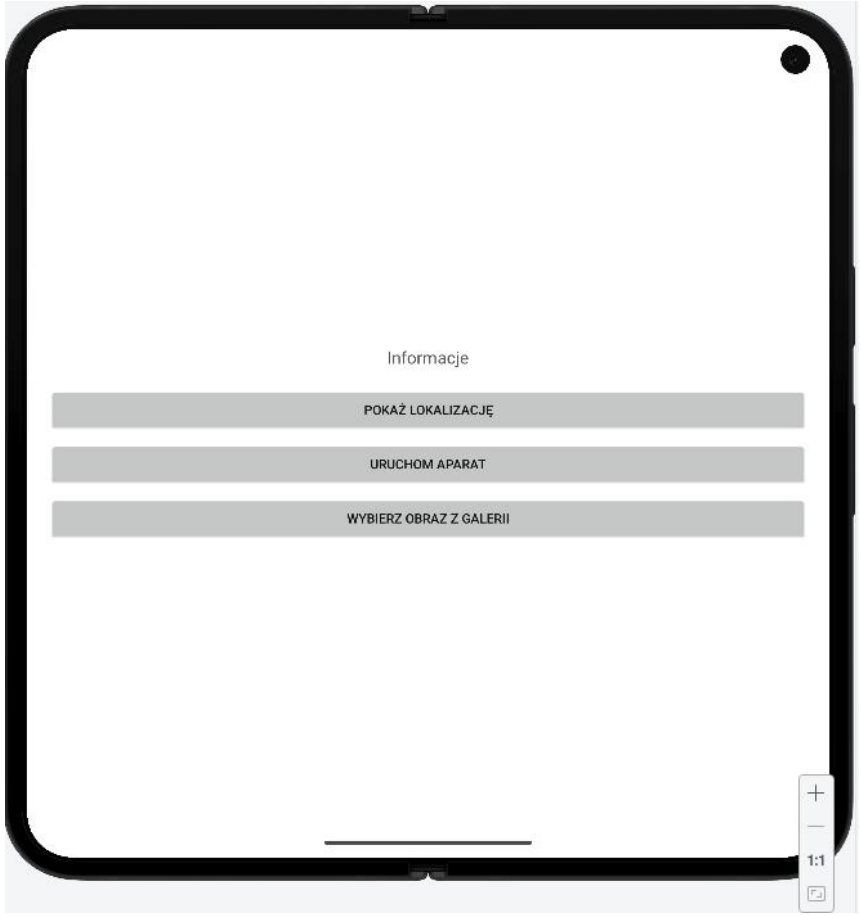
# Pliki XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:gravity="center"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="24dp">

  <TextView
    android:id="@+id/infoText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Informacje"
    android:textSize="18sp"
    android:layout_marginBottom="20dp" />

  <Button
    android:id="@+id/btnLocation"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Pokaż lokalizację" />

  <Button
    android:id="@+id/btnCamera"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Uruchom aparat"
    android:layout_marginTop="10dp" />

  <Button
    android:id="@+id/btnStorage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Wybierz obraz z galerii"
    android:layout_marginTop="10dp" />
</LinearLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.uprawnienia">

  <!-- Wymagane uprawnienia -->
  <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
  <uses-permission
android:name="android.permission.CAMERA" />
  <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />

  <application
    android:allowBackup="true"
    android:label="UprawnieniaApp"
    android:icon="@mipmap/ic_launcher"

android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.uprawnienia">

    <!-- MainActivity z intent-filter oraz
android:exported="true" -->
    <activity
      android:name=".MainActivity"
      android:exported="true">
      <intent-filter>
        <action
android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

  </application>

</manifest>
```



# Efekt działania




Aplikacja po prostu daje dostęp do zasobów systemu i urządzeń, które pozwalają na uruchomienie określonych usług.

Jeżeli udostępnimy w ustawieniach telefonu konkretne opcje – dostęp do plików, aparatu czy lokalizacji to i tak aplikacja zapyta Nas czy chcemy zatem uprawnienia w kodzie i tak są konieczne.

# Klasa Activity

Activity to podstawowa jednostka interfejsu użytkownika w Androidzie.
Każdy ekran aplikacji to osobna aktywność.
Klasa Activity pochodzi z pakietu android.app.

onCreate() – metoda wywoływana przy uruchomieniu aktywności.
setContentView() – ustawia layout XML dla ekranu.

```
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);    }}
```

# Cykl życia Activity

onCreate() – inicjalizacja, tworzenie widoku.
onStart() – aktywność widoczna, ale nieaktywna.
onResume() – aktywność aktywna i gotowa do interakcji.
onPause() – aktywność traci fokus.
onStop() – aktywność niewidoczna.
onDestroy() – aktywność niszczona.

**Intent – służy do przechodzenia pomiędzy Activity**

Intent intent = new Intent(CurrentActivity.this, NextActivity.class);startActivity(intent);

# Intent

Intent pozwala przechodzić między ekranami.
Można przesyłać dane (putExtra / getIntent().getStringExtra()).

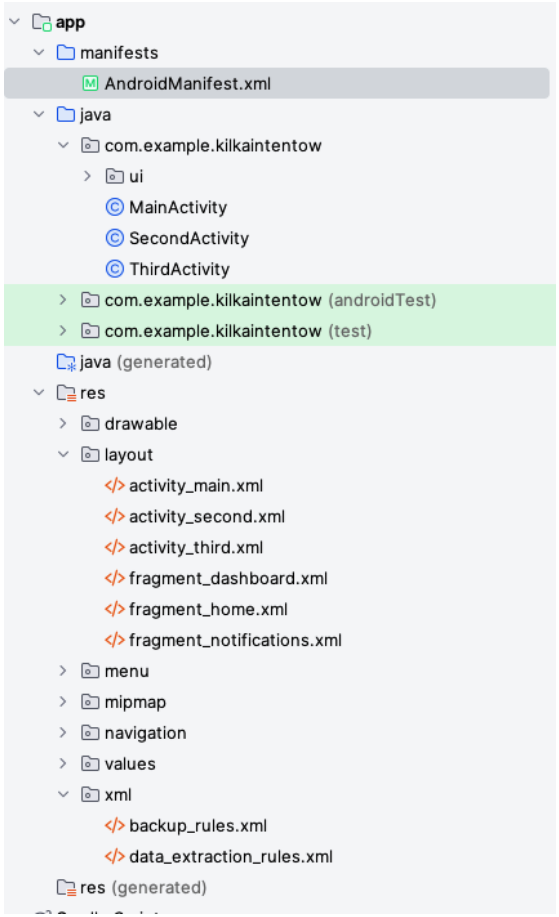
```
Intent intent = new Intent(this, SecondActivity.class);
intent.putExtra("name", "Jan");startActivity(intent);
```

**Praktyczne zastosowania Activity:**
- Formularze, quizy, rejestracja użytkownika.
- Lista produktów i szczegóły produktu.
- Nawigacja po aplikacji.

```
String name = getIntent().getStringExtra("name");
```

**Często stosowane metody:**
- finish() – zamyka bieżącą aktywność.
- getIntent() – pobiera intencję wywołującą aktywność.
- onBackPressed() – zachowanie przy naciśnięciu przycisku Wstecz.
- startActivityForResult() – uruchamianie z oczekiwaniem na wynik.

# Przykład zastosowania klas Activity oraz Intent




Struktura projektu

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/re
s/android"
  package="com.example.kilkaintentow">

  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="MultiScreenApp"

android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Kilkaintentow">

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

    <!-- MainActivity with an explicit exported
attribute -->
    <activity android:name=".MainActivity"
      android:exported="true">
      <intent-filter>
        <action
android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCHER"
/>
      </intent-filter>
    </activity>

  </application>

</manifest>
```

Manifest

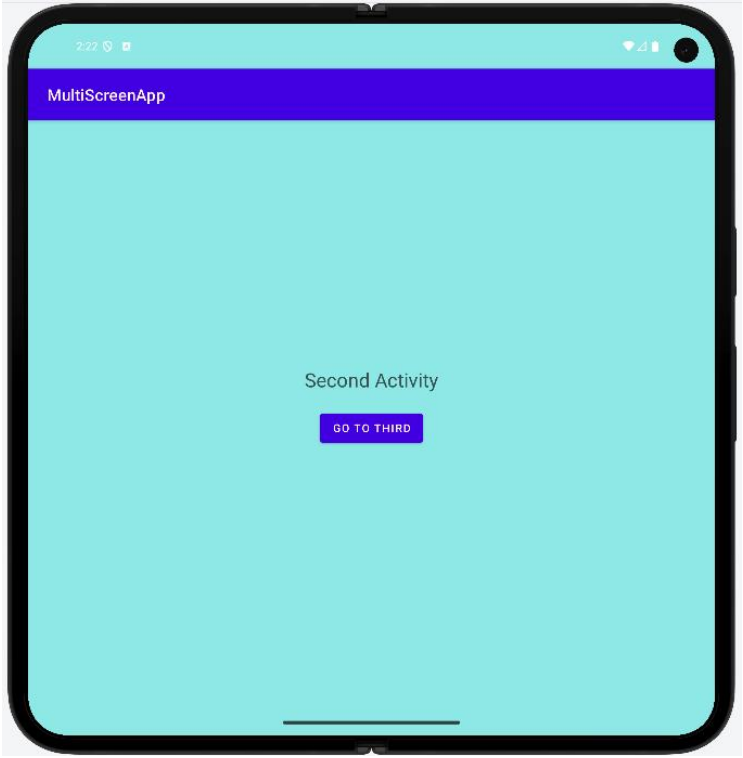
# Klasy Activity

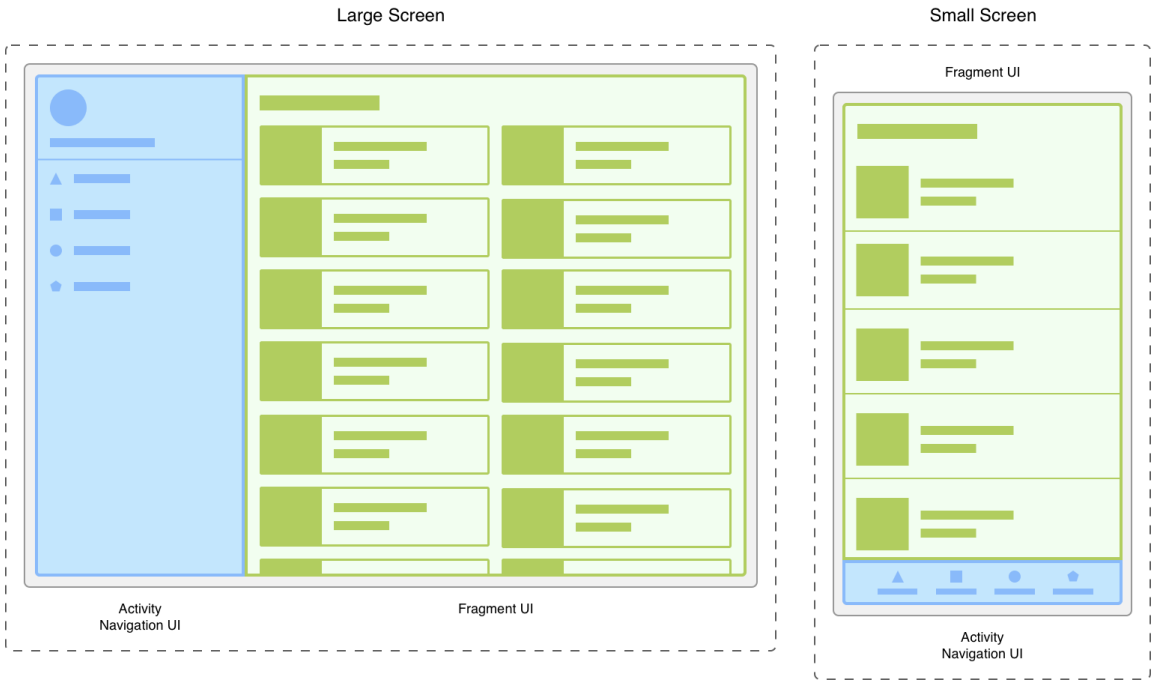
```
package com.example.kilkaintentow;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import
androidx.appcompat.app.AppCompatActi
vity;

public class MainActivity extends
AppCompatActivity {
   Button btnGoToSecond;

   @Override
   protected void onCreate(Bundle
savedInstanceState) {
      super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

      btnGoToSecond =
findViewById(R.id.btnToSecond);
      btnGoToSecond.setOnClickListener(v
-> {
         Intent intent = new
Intent(MainActivity.this,
SecondActivity.class);
         startActivity(intent);
      });
   }
}
```

```
package
com.example.kilkaintentow;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import
androidx.appcompat.app.AppCo
mpatActivity;

public class SecondActivity
extends AppCompatActivity {
   Button btnGoToThird;

   @Override
   protected void onCreate(Bundle
savedInstanceState) {

super.onCreate(savedInstanceSta
te);

setContentView(R.layout.activity_
second);

      btnGoToThird =
findViewById(R.id.btnToThird);

btnGoToThird.setOnClickListener(
v -> {
         Intent intent = new
Intent(SecondActivity.this,
ThirdActivity.class);
         startActivity(intent);
      });
   }
}
```

```
package
com.example.kilkaintentow;


import android.content.Intent;

import android.os.Bundle;

import android.widget.Button;

import
androidx.appcompat.app.App
CompatActivity;


public class SecondActivity
extends AppCompatActivity {

   Button btnGoToThird;


   @Override

   protected void
onCreate(Bundle
savedInstanceState) {


super.onCreate(savedInstanc
eState);

setContentView(R.layout.activ
ity_second);


      btnGoToThird =
findViewById(R.id.btnToThird);


btnGoToThird.setOnClickListe
ner(v -> {

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

         startActivity(intent);

      });

   }

}
```

# Wygląd i działanie aplikacji


# Fragments

Fragment to **modułowy komponent interfejsu użytkownika**.

Reprezentuje **część aktywności (Activity)**.

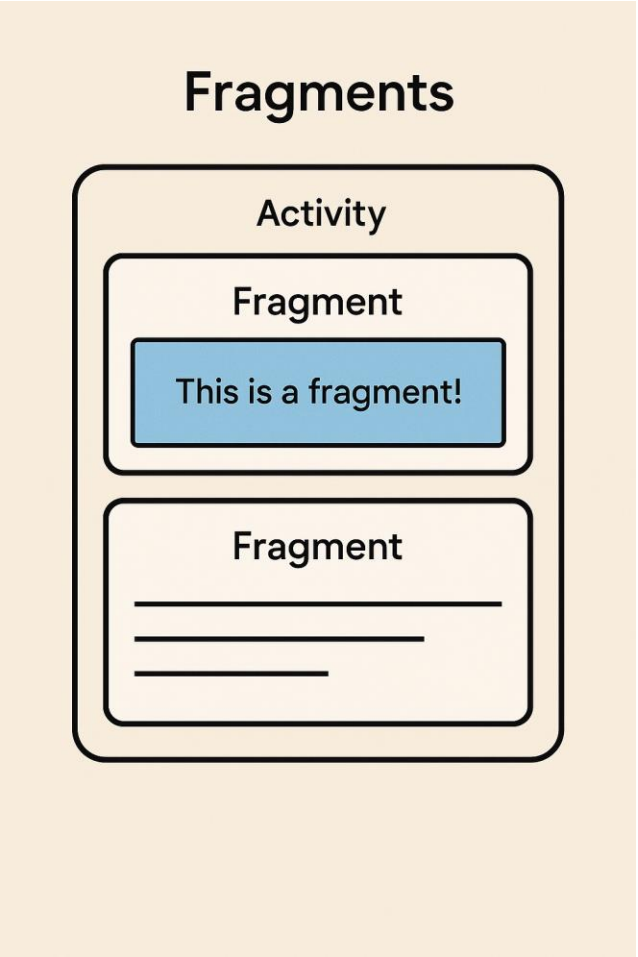
Może zawierać własny layout i logikę.

Fragmenty można dynamicznie dodawać, usuwać, wymieniać.

# Fragments

**Dlaczego warto?**

- Lepsza organizacja kodu.
- Możliwość **ponownego użycia widoków**.
- Ułatwia projektowanie aplikacji na tablety i telefony (wielopanelowe UI).
- Obsługa **nawigacji między widokami** w jednej aktywności.



# Fragments

## Struktura:

```
public class MyFragment extends Fragment {
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
            return inflater.inflate(R.layout.fragment_my, container, false);
             }
}
```

Plik JAVA

Plik XML

```
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
 android:text="To jest fragment!" />
```

# Fragments

**Statyczne dodawanie Fragmentów do Activity:**

```
<fragment    android:name="com.example.MyFragment"
android:id="@+id/myFragment"
android:layout_width="match_parent"
android:layout_height="wrap_content" />
```

**Dodawanie dynamiczne w kodzie:**

```
FragmentManager fragmentManager = getSupportFragmentManager();
FragmentTransaction transaction = fragmentManager.beginTransaction();
transaction.replace(R.id.fragmentContainer, new MyFragment());
transaction.commit();
```

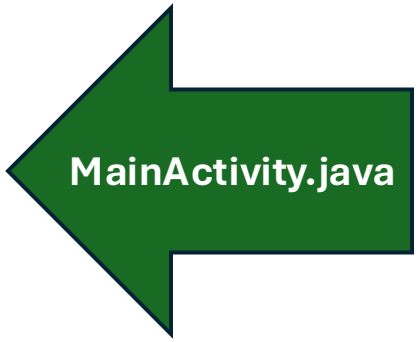
# Fragments

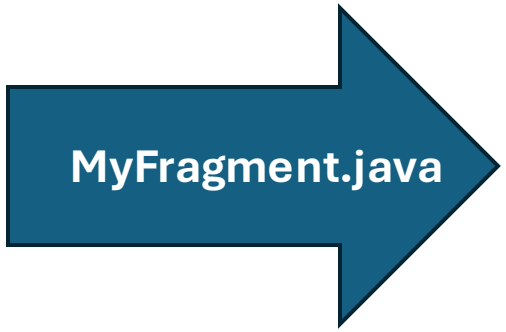
```
<FrameLayout    android:id="@+id/fragmentContainer"
android:layout_width="match_parent"
android:layout_height="match_parent"/>
```

Fragment to nie tylko sposób na uporządkowanie wyglądu aplikacji. Jeżeli wiemy, że może przechowywać formularz, listę, tabelę lub inne widżety przechowujące dane to znaczy, że będzie nośnikiem dla tych danych. Dane będą mogły być przekazywane z jednego fragmentu do drugiego.

# Przekazywanie danych do fragmentów

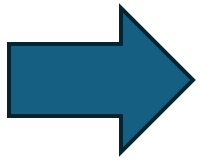
```
MyFragment fragment = new MyFragment();
Bundle bundle = new Bundle();
bundle.putString("username", "Jan");
fragment.setArguments(bundle);
```



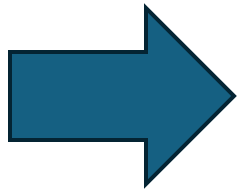




```
String user = getArguments().getString("username");
```

# Komunikacja Fragment-Activity

**Fragment**  **Activity**

```
((MainActivity) getActivity()).somePublicMethod();
```

**Activity**  **Fragment**

```
MyFragment fragment = (MyFragment) getSupportFragmentManager()
.findFragmentById(R.id.myFragment);

fragment.updateUI();
```

# Cykl życia Fragmentu

| METODA | OPIS DZIAŁANIA |
| --- | --- |
| onAttach(Context) | Wywoływana, gdy fragment zostaje **przypięty do aktywności**. Tu można uzyskać kontekst. |
| onCreate(Bundle) | Inicjalizacja fragmentu – np. tworzenie zmiennych, odczyt argumentów z Bundle. |
| onCreateView(...) | Tworzenie i **zwrócenie widoku** (layoutu) fragmentu. Tutaj "nadmuchuje się" XML. |
| onViewCreated(...) | Widok został już utworzony – można np. ustawiać **listenery**, inicjalizować komponenty UI. |
| onActivityCreated(...) | Aktywność jest już gotowa – można komunikować się z aktywnością. |
| onStart() | Fragment **staje się widoczny** dla użytkownika. |
| onResume() | Fragment **wchodzi w interakcję** z użytkownikiem (jest na wierzchu). |
| onPause() | Fragment **traci fokus**, ale nadal jest widoczny. Zatrzymaj np. animacje. |
| onStop() | Fragment **nie jest już widoczny**. Można zwolnić zasoby. |
| onDestroyView() | Usuwany jest widok fragmentu – warto posprzątać zasoby GUI. |
| onDestroy() | Fragment jest niszczony – usuń zasoby niezwiązane z widokiem. |
| onDetach() | Fragment jest **odczepiany od aktywności**. Nie ma już dostępu do Context. |

# Przykładowa aplikacja z Fragments

Po naciśnięciu górnej części z napisem zmienia się kolor części dolnej oraz wyświetla się komunikat systemowy

# Pliki JAVA

```
package com.example.fragmenty;

import android.os.Bundle;
import android.widget.Button;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;

public class MainActivity extends AppCompatActivity {

    Button redButton;

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

        redButton = findViewById(R.id.buttonRed);

        redButton.setOnClickListener(v -> {
            Toast.makeText(this, "Klik działa", Toast.LENGTH_SHORT).show();
            loadFragment(ColorFragment.newInstance("#FF0000"));
        });

        loadFragment(ColorFragment.newInstance("#00FF00")); // startowy fragment
    }

    private void loadFragment(Fragment fragment) {
        getSupportFragmentManager().beginTransaction()
                .replace(R.id.fragmentContainer, fragment)
                .commit();
    }
}
```

```
package com.example.fragmenty;

import android.graphics.Color;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

public class ColorFragment extends
Fragment {

    private static final String ARG_COLOR =
"color";

    public static ColorFragment
newInstance(String color) {
        ColorFragment fragment = new
ColorFragment();
        Bundle args = new Bundle();
        args.putString(ARG_COLOR, color);
        fragment.setArguments(args);
        return fragment;
    }

    public ColorFragment() {}

    @Override
    public View onCreateView(LayoutInflater
inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view =
inflater.inflate(R.layout.fragment_color,
container, false);

        if (getArguments() != null) {
            String color =
getArguments().getString(ARG_COLOR);
            try {

view.setBackgroundColor(Color.parseColor(
color));
                TextView text =
view.findViewById(R.id.textViewColor);
                text.setText("Kolor: " + color);
            } catch (IllegalArgumentException e) {

view.setBackgroundColor(Color.LTGRAY);
            }
        }

        return view;
    }
}
```

# Pliki XML

Plik XML Fragmentu może zawierać własny Layout, któremu odpowiada kod napisany w języku JAVA lub Kotlin.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical" android:layout_width="match_parent"
  android:layout_height="match_parent">

  <Button
    android:id="@+id/buttonRed"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Czerwony"
    android:textSize="45dp"
    android:paddingTop="120dp"
    android:layout_gravity="center"
    android:layout_margin="@dimen/activity_horizontal_margin"/>

  <FrameLayout
    android:id="@+id/fragmentContainer"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1" />
</LinearLayout>
```

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/colorFragmentLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#00FF00">

    <TextView
        android:id="@+id/textViewColor"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Kolor"
        android:textColor="#000000"
        android:layout_gravity="center"
        android:textSize="24sp" />
</FrameLayout>
```

# Adaptery

**Adapter** to "tłumacz", który zamienia dane (np. lista obiektów Pokój) na widoki (View), które Android może wyświetlić na ekranie w postaci listy, siatki lub innego układu.

| Adapter | Gdzie używany | Opis |
|---|---|---|
| ArrayAdapter | ListView / Spinner | Prosty adapter dla listy tekstów. |
| BaseAdapter | ListView, GridView | Bardziej elastyczny, ale wymaga więcej kodu. |
| RecyclerView.Adapter | RecyclerView | Najczęściej używany do dynamicznych list – szybki i nowoczesny. |
| SimpleCursorAdapter | ListView z bazą danych | Stary adapter do danych z SQLite. |

# Przykład Adaptera

```
public class RoomAdapter extends RecyclerView.Adapter<RoomAdapter.RoomViewHolder> {

    private List<Room> roomList;

    public RoomAdapter(List<Room> roomList) {
        this.roomList = roomList;
    }

    @NonNull
    @Override
    public RoomViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.room_item, parent, false);
        return new RoomViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull RoomViewHolder holder, int position) {
        Room room = roomList.get(position);
        holder.roomName.setText(room.getName());
        // itd...
    }

    @Override
    public int getItemCount() {
        return roomList.size();
    }

    public static class RoomViewHolder extends RecyclerView.ViewHolder {
        TextView roomName;

        public RoomViewHolder(View itemView) {
            super(itemView);
            roomName = itemView.findViewById(R.id.roomName);
        }
    }
}
```

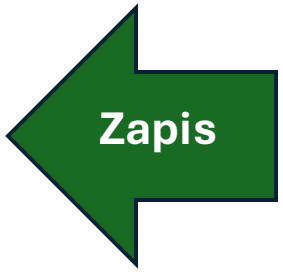
**Co musisz zrozumieć:**

- Adapter tworzy **ViewHoldery** (czyli widoki jednego elementu)
- Wiąże dane z widokiem (onBindViewHolder)
- getItemCount() mówi ile elementów ma lista
- Adaptery pozwalają oddzielić logikę danych od prezentacji.

# Shared Preferences – zapisywanie danych

Służy do zapisywania danych typu klucz-wartość.
Idealne do: ustawień, loginów, stanu aplikacji.

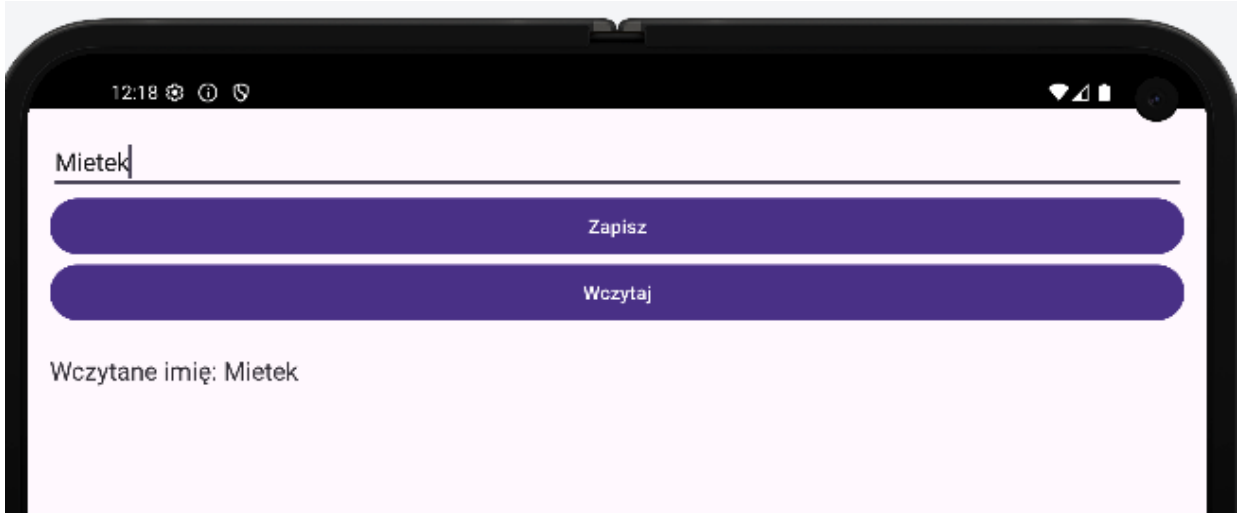
```
SharedPreferences prefs = getSharedPreferences("MyPrefs",
MODE_PRIVATE);SharedPreferences.Editor editor =
prefs.edit();editor.putString("user", "Jan");editor.apply();
```





```
SharedPreferences prefs = getSharedPreferences("MyPrefs",
MODE_PRIVATE);String user = prefs.getString("user", "Brak");
```

# Przykładowa aplikacja



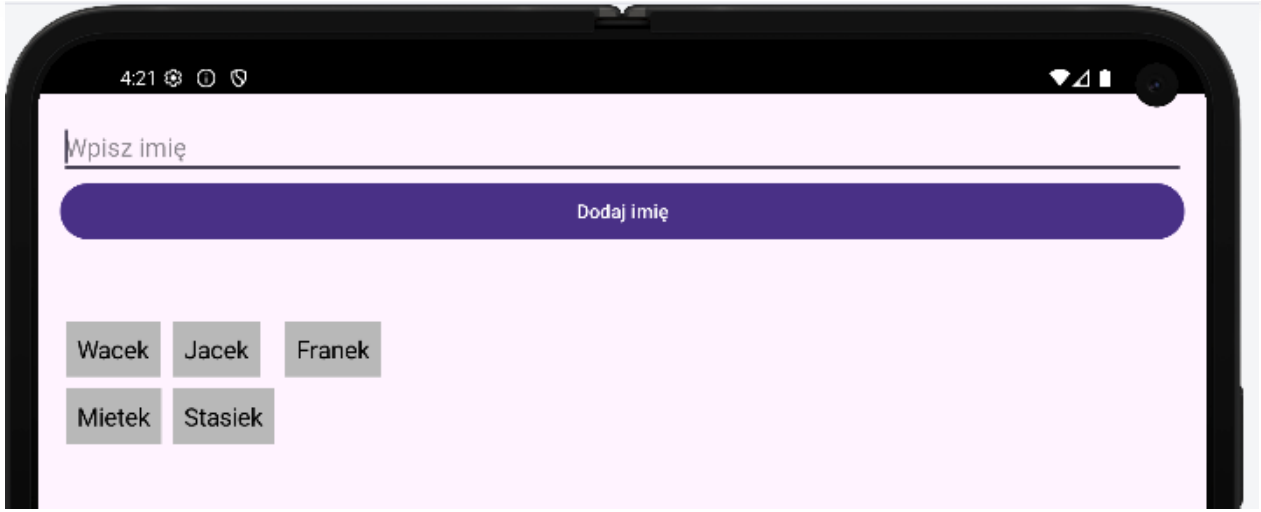

Aplikacja pobiera dane od użytkownika a następnie zapisuje je w pamięci. Po naciśnięciu przycisku „Wczytaj” dane są wyświetlane na liście.

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

xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:padding="16dp"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <EditText
    android:id="@+id/editTextName"
    android:layout_width="match_parent"

android:layout_height="wrap_content"
    android:hint="Wpisz swoje imię"/>

  <Button
    android:id="@+id/buttonSave"
    android:layout_width="match_parent"

android:layout_height="wrap_content"
    android:text="Zapisz"/>

  <Button
    android:id="@+id/buttonLoad"
    android:layout_width="match_parent"

android:layout_height="wrap_content"
    android:text="Wczytaj"/>

  <TextView
    android:id="@+id/textViewResult"
    android:layout_width="match_parent"

android:layout_height="wrap_content"
    android:text="Tutaj pojawi się
zapisane imię"
    android:paddingTop="20dp"
    android:textSize="18sp"/>

</LinearLayout>
```

# Przykładowa aplikacja

```
package com.example.preferencje;

import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

import com.example.preferencje.R;

public class MainActivity extends AppCompatActivity {

    private EditText editTextName;
    private TextView textViewResult;
    private Button buttonSave, buttonLoad;

    private static final String PREFS_NAME = "MyPrefsFile";
    private static final String KEY_NAME = "user_name";

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

        editTextName = findViewById(R.id.editTextName);
        textViewResult = findViewById(R.id.textViewResult);
        buttonSave = findViewById(R.id.buttonSave);
        buttonLoad = findViewById(R.id.buttonLoad);

        buttonSave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                // Pobranie SharedPreferences
                SharedPreferences sharedPreferences =
getSharedPreferences(PREFS_NAME, MODE_PRIVATE);

                // Otworzenie edytora
                SharedPreferences.Editor editor = sharedPreferences.edit();

                // Zapisanie tekstu pod kluczem
                editor.putString(KEY_NAME, editTextName.getText().toString());

                // Zatwierdzenie zmian
                editor.apply();

                textViewResult.setText("Imię zapisane!");
            }
        });

        buttonLoad.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                SharedPreferences sharedPreferences =
getSharedPreferences(PREFS_NAME, MODE_PRIVATE);

                // Odczyt wartości (domyślnie pusty string jeśli brak)
                String name = sharedPreferences.getString(KEY_NAME, "");

                if(name.isEmpty()) {
                    textViewResult.setText("Brak zapisanego imienia");
                } else {
                    textViewResult.setText("Wczytane imię: " + name);
                }
            }
        });
    }
}
```

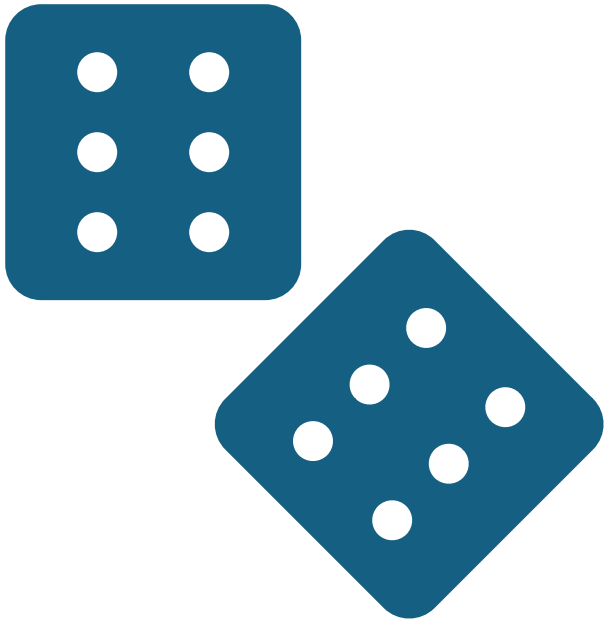
# Zapis danych w układzie 3x3




Użytkownik wpisuje dane a następnie są one grupowane wewnątrz siatki. To doskonały przykład, który można rozbudować jako grę .

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

xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:padding="16dp"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <EditText
    android:id="@+id/editTextName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Wpisz imię" />

  <Button
    android:id="@+id/buttonAdd"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Dodaj imię" />

  <TextView
    android:id="@+id/textViewMessage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text=""
    android:textColor="#FF0000"
    android:paddingTop="10dp"
    android:textSize="16sp" />

  <GridLayout
    android:id="@+id/gridLayoutNames"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:columnCount="3"
    android:rowCount="3"
    android:paddingTop="20dp"
    android:alignmentMode="alignMargins"
    android:useDefaultMargins="true" />

</LinearLayout>
```

# Plik JAVA



```
package com.example.bingo;

import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.GridLayout;
import android.widget.TextView;
import android.graphics.Color;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private EditText editTextName;
    private Button buttonAdd;
    private GridLayout gridLayoutNames;
    private TextView textViewMessage;

    private static final String PREFS_NAME = "MyPrefsFile";
    private static final String KEY_NAMES = "names_list";
    private List<String> namesList = new ArrayList<>();

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

        editTextName = findViewById(R.id.editTextName);
        buttonAdd = findViewById(R.id.buttonAdd);
        gridLayoutNames = findViewById(R.id.gridLayoutNames);
        textViewMessage = findViewById(R.id.textViewMessage);

        loadNames();
        updateGrid();

        buttonAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String newName = editTextName.getText().toString().trim();

                if (TextUtils.isEmpty(newName)) {
                    textViewMessage.setText("Wpisz imię!");
                    return;
                }

                if (namesList.size() >= 9) {
                    textViewMessage.setText("Za dużo wpisów!");
                    return;
                }

                namesList.add(newName);
                saveNames();
                updateGrid();
                editTextName.setText("");
                textViewMessage.setText(""); // Wyczyść komunikat
            }
        });
    }

    private void loadNames() {
        SharedPreferences sharedPreferences = getSharedPreferences(PREFS_NAME,
MODE_PRIVATE);
        String savedNames = sharedPreferences.getString(KEY_NAMES, "");

        if (!savedNames.isEmpty()) {
            namesList = new ArrayList<>(Arrays.asList(savedNames.split(",")));
        }
    }

    private void saveNames() {
        SharedPreferences sharedPreferences = getSharedPreferences(PREFS_NAME,
MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();

        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < namesList.size(); i++) {
            sb.append(namesList.get(i));
            if (i != namesList.size() - 1) {
                sb.append(",");
            }
        }

        editor.putString(KEY_NAMES, sb.toString());
        editor.apply();
    }

    private void updateGrid() {
        gridLayoutNames.removeAllViews();

        for (String name : namesList) {
            TextView tv = new TextView(this);
            tv.setText(name);
            tv.setTextSize(18);
            tv.setTextColor(Color.BLACK);
            tv.setBackgroundColor(Color.LTGRAY);
            tv.setPadding(20, 20, 20, 20);

            GridLayout.LayoutParams param = new GridLayout.LayoutParams();
            param.width = GridLayout.LayoutParams.WRAP_CONTENT;
            param.height = GridLayout.LayoutParams.WRAP_CONTENT;
            param.setMargins(10, 10, 10, 10);
            tv.setLayoutParams(param);

            gridLayoutNames.addView(tv);
        }
    }
}
```

# Content Resolver – odczyt danych systemowych

Umożliwia dostęp do danych z ContentProviderów (kontakty, media).
Przykładowe ContentProvidery: ContactsContract, MediaStore.


```
<uses-permission
android:name="android.permi
ssion.READ_CONTACTS" />
```

```
Cursor cursor = getContentResolver().query(
ContactsContract.Contacts.CONTENT_URI,
 null, null, null, null);
if (cursor != null && cursor.moveToFirst()) {
String name = cursor.getString(
cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME))
;
Log.d("Kontakt", name);    cursor.close();
}
```

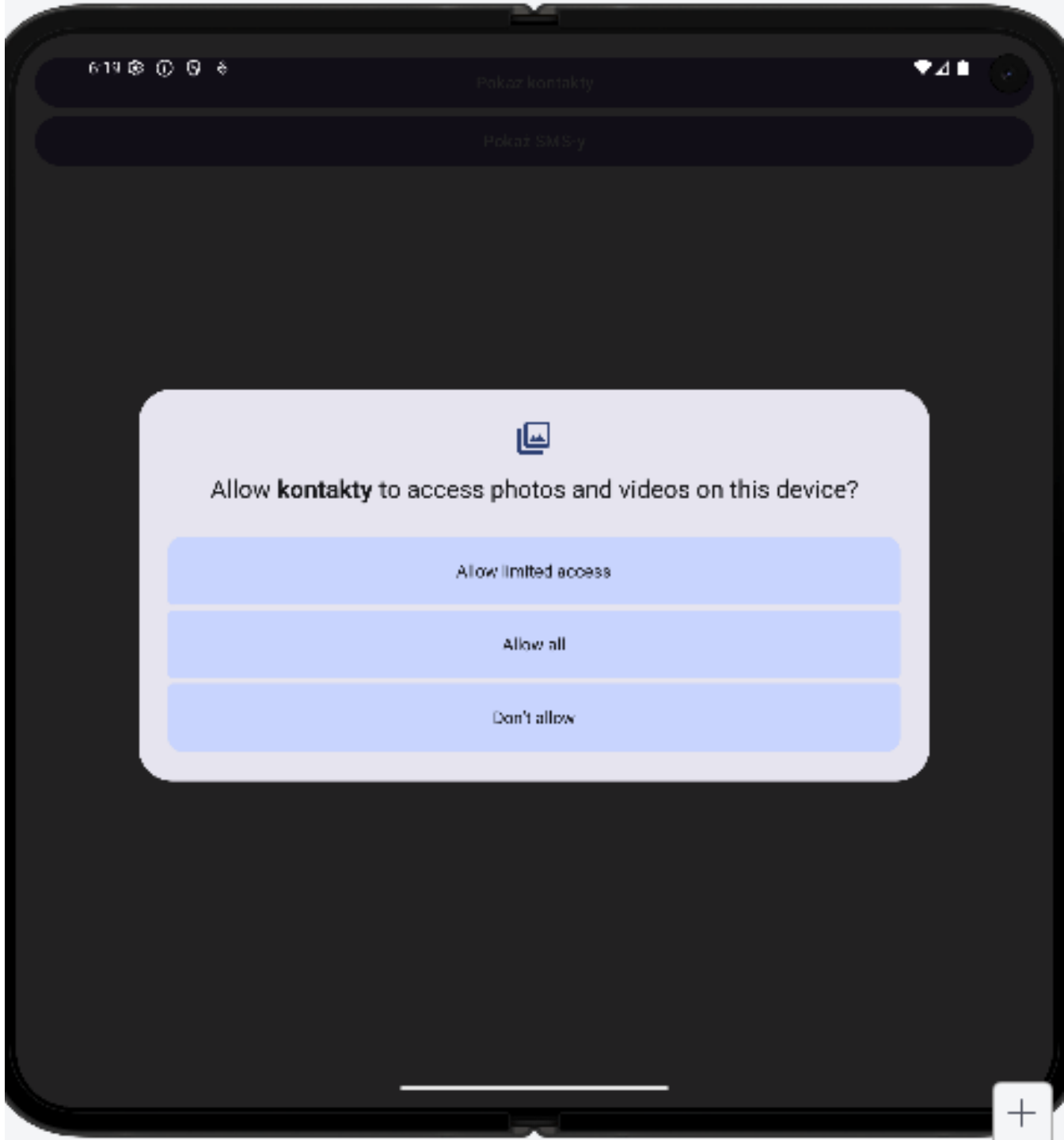
# Content Resolver – odczytywanie danych systemowych

finish() – zamknięcie aktywności.

getIntent() – pobranie danych z Intent.

onBackPressed() – własne działanie przy wstecz.

startActivityForResult() – uruchomienie z oczekiwaniem na rezultat.

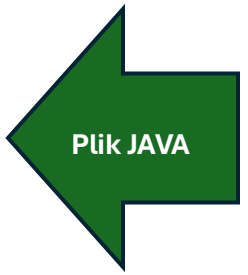


# Odczytanie danych z telefonu

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.kontakty">

    <!-- Uprawnienia -->
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
        android:maxSdkVersion="32"/>
    <uses-permission android:name="android.permission.READ_MEDIA_IMAGES"/>

    <application
        android:allowBackup="true"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.Kontakty">
        <activity android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

# Odczytywanie danych z telefonu

```
package com.example.kontakty;

import android.Manifest;
import android.content.ContentResolver;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    private static final int PERMISSION_REQUEST_CODE
= 123;
    private static final String[] permissions = {
            Manifest.permission.READ_CONTACTS,
            Manifest.permission.READ_SMS,

Manifest.permission.READ_EXTERNAL_STORAGE,
            Manifest.permission.READ_MEDIA_IMAGES
    };

    ListView listView;
    Button btnLoadContacts, btnLoadSMS;

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

        listView = findViewById(R.id.listView);
        btnLoadContacts =
findViewById(R.id.btnLoadContacts);
        btnLoadSMS = findViewById(R.id.btnLoadSMS);

        if (!hasPermissions()) {
            ActivityCompat.requestPermissions(this,
permissions, PERMISSION_REQUEST_CODE);
        }

        btnLoadContacts.setOnClickListener(v -> {
            if (hasPermissions()) {
                showContacts();
            } else {
                Toast.makeText(this, "Brak uprawnień do
kontaktów", Toast.LENGTH_SHORT).show();
            }
        });

        btnLoadSMS.setOnClickListener(v -> {
            if (hasPermissions()) {
                showSMS();
            } else {
                Toast.makeText(this, "Brak uprawnień do
SMS", Toast.LENGTH_SHORT).show();
            }
        });
    }

    private boolean hasPermissions() {
        for (String permission : permissions) {
            if (ContextCompat.checkSelfPermission(this,
permission)
                    != PackageManager.PERMISSION_GRANTED)
{
                return false;
            }
        }
        return true;
    }

    private void showContacts() {
        ArrayList<String> list = new ArrayList<>();
        ContentResolver cr = getContentResolver();

        Cursor cursor =
cr.query(ContactsContract.CommonDataKinds.Phone
.CONTENT_URI,
                null, null, null, null);

        if (cursor != null) {
            while (cursor.moveToNext()) {
                String name = cursor.getString(

cursor.getColumnIndex(ContactsContract.CommonD
ataKinds.Phone.DISPLAY_NAME));
                String phone = cursor.getString(

cursor.getColumnIndex(ContactsContract.CommonD
ataKinds.Phone.NUMBER));
                list.add(name + ": " + phone);
            }
            cursor.close();
        }

        listView.setAdapter(new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, list));
    }

    private void showSMS() {
        ArrayList<String> smsList = new ArrayList<>();
        ContentResolver cr = getContentResolver();

        Uri uri = Uri.parse("content://sms/inbox");
        Cursor cursor = cr.query(uri, null, null, null, null);

        if (cursor != null) {
            while (cursor.moveToNext()) {
                String address =
cursor.getString(cursor.getColumnIndex("address"));
                String body =
cursor.getString(cursor.getColumnIndex("body"));
                smsList.add("Od: " + address + "\n" + body);
            }
            cursor.close();
        }

        listView.setAdapter(new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, smsList));
    }

    @Override
    public void onRequestPermissionsResult(int
requestCode, @NonNull String[] permissions,
                                           @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);
        if (requestCode == PERMISSION_REQUEST_CODE)
{
            if (hasPermissions()) {
                Toast.makeText(this, "Uprawnienia przyznane",
Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(this, "Brak wymaganych
uprawnień!", Toast.LENGTH_LONG).show();
            }
        }
    }
}
```

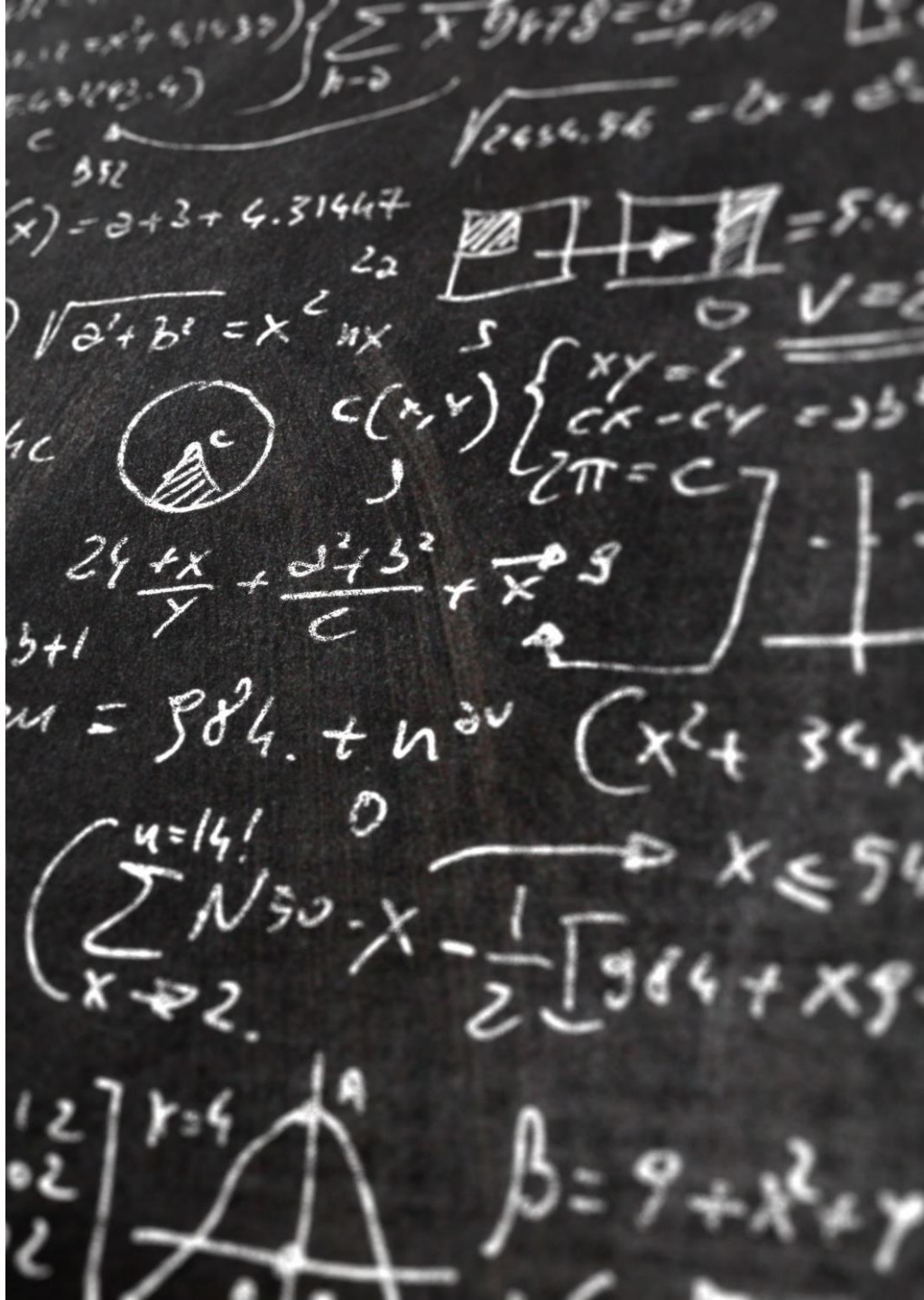



# Odczytywanie danych z telefonu

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="16dp"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/btnLoadContacts"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Pokaż kontakty" />

    <Button
        android:id="@+id/btnLoadSMS"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Pokaż SMS-y" />

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"/>
</LinearLayout>
```

# Podsumowanie - pytania

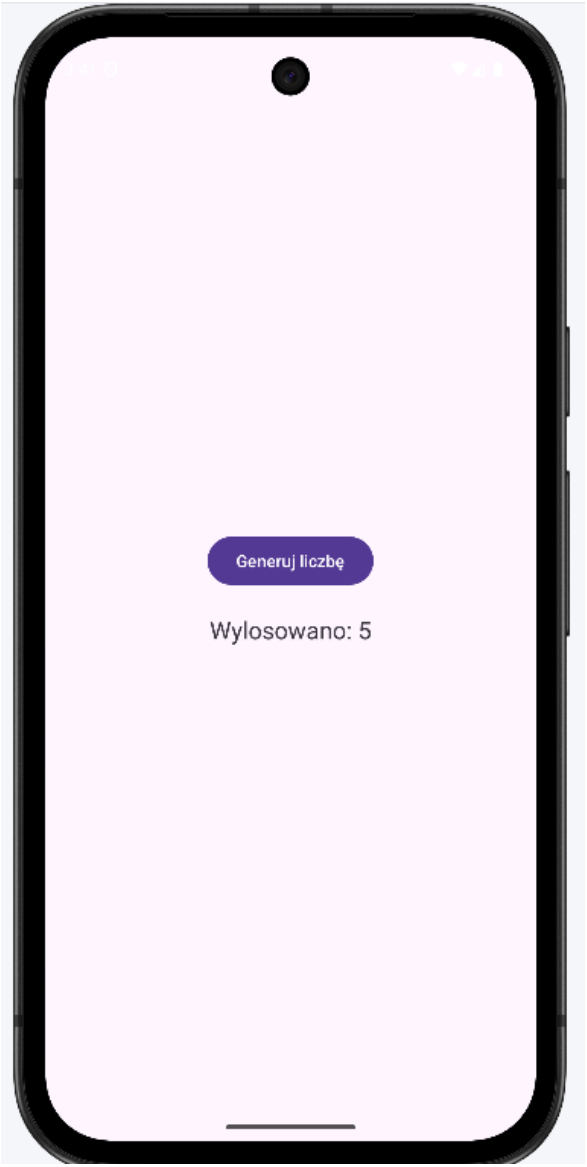
1. Wymień przynajmniej 5 właściwości dowolnego Widżeta
2. Wymień znane Tobie rodzaje layout
3. Wymień przynajmniej 5 uprawnień w Android
4. Co przechowuje klasa Activity?
5. Co to jest Intent i za co odpowiada?
6. Za co odpowiadają Shared Preferences?
7. Jakie zasoby pozwala odczytać Content Resolver?
8. Dlaczego Fragments porównuje się do znaczników HTML i CSS?



# Zadania do samodzielnego wykonania

## Zadanie 1

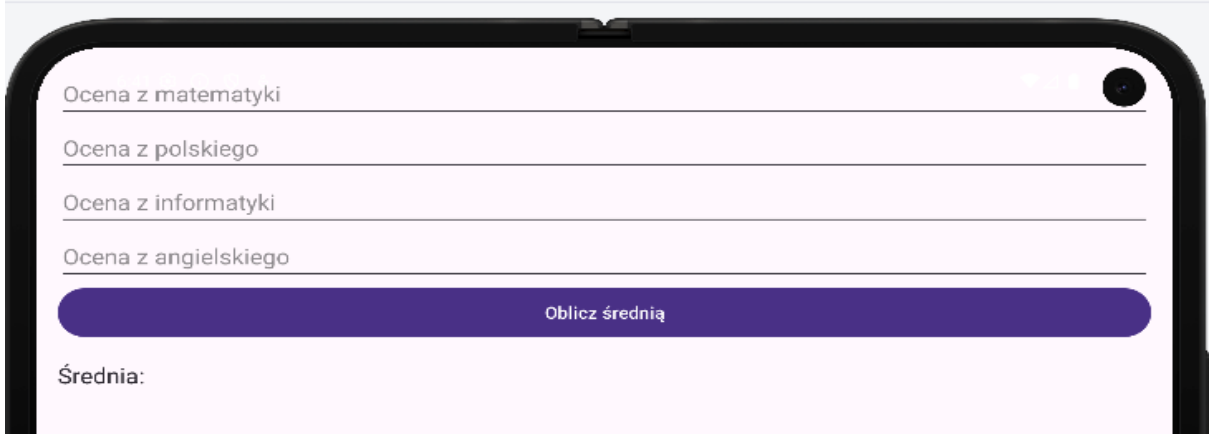
Aplikacja losuje liczby z przedziału od 1 do 100. Wynik zostanie wyświetlony na etykiecie.

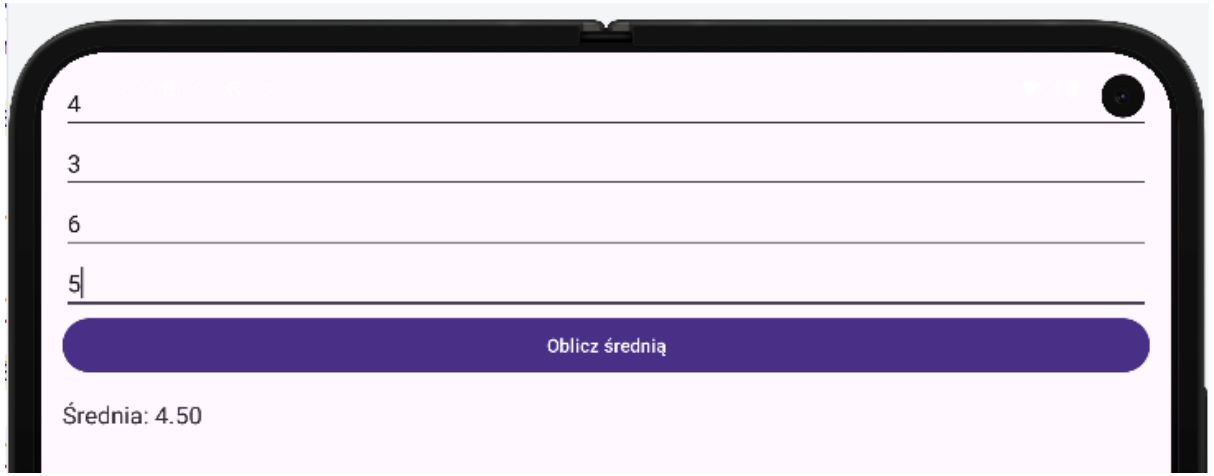


# Zadania do samodzielnego wykonania

## Zadanie 2

Aplikacja liczy średnią arytmetyczną z kilku przedmiotów.



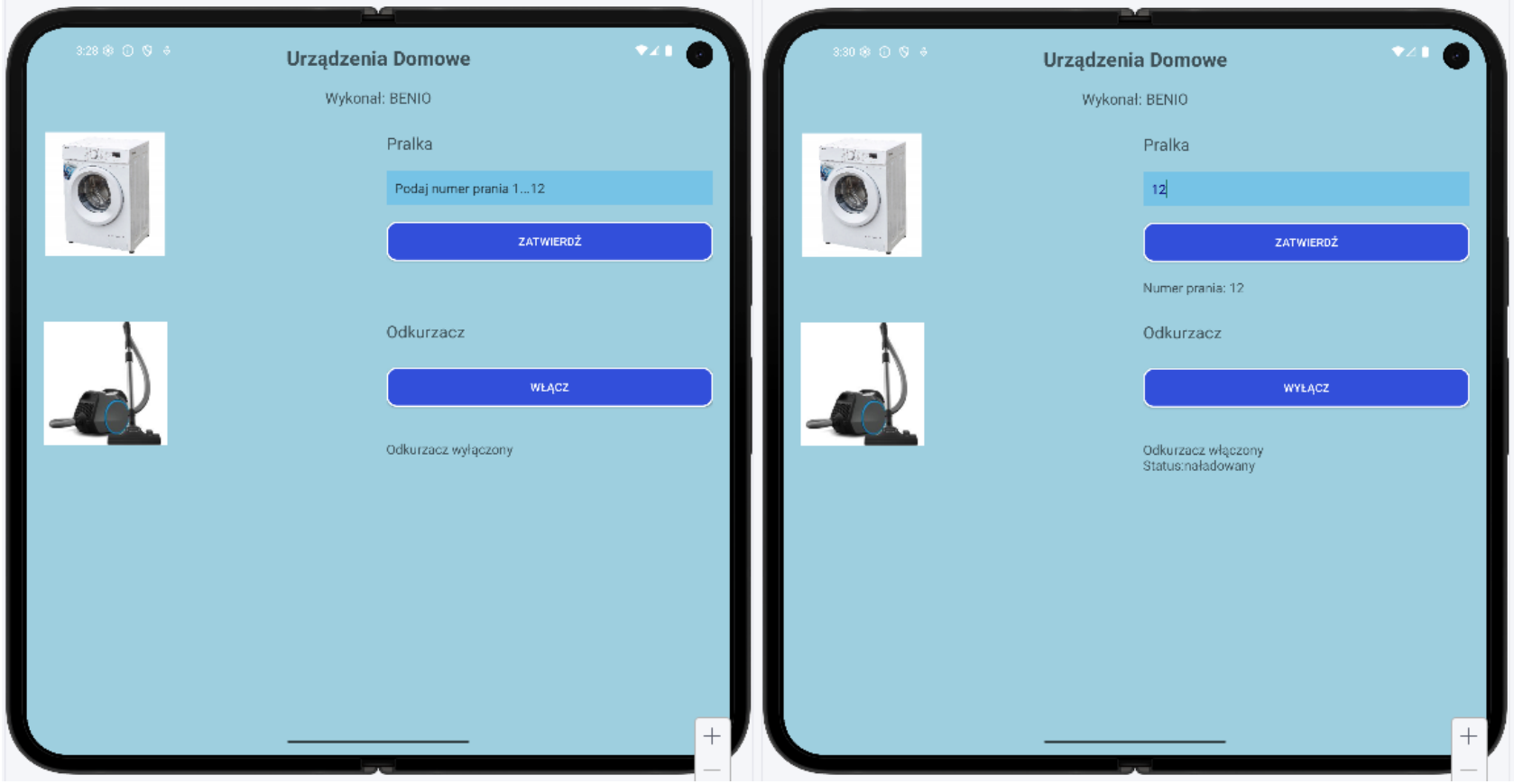



# Zadania do samodzielnego wykonania

## Zadanie 3

Użytkownik podaje numer. Jeżeli jest większy niż 12 to program prosi o podanie właściwego numeru. Przycisk „Zatwierdź” wyświetla etykietę.

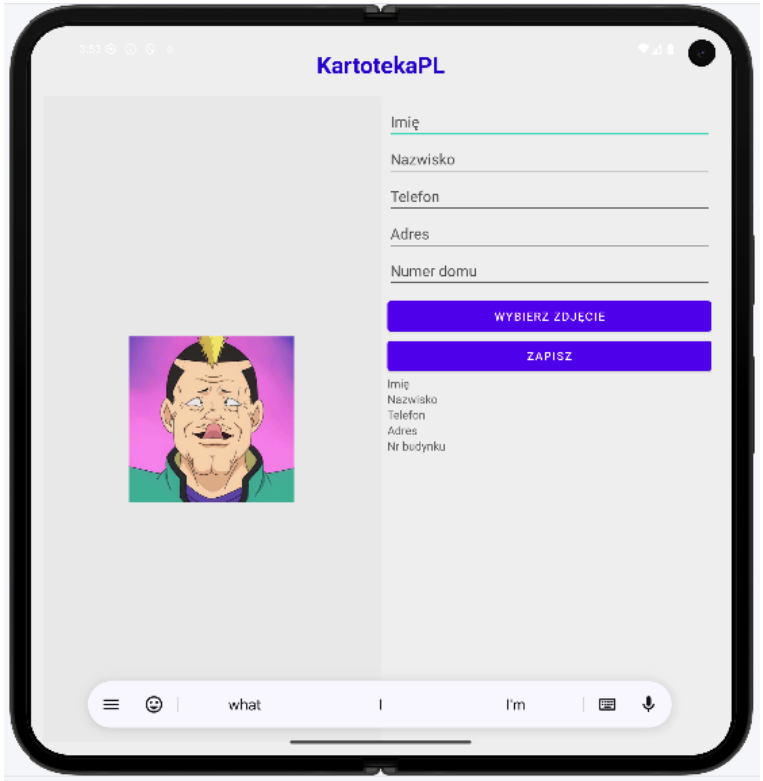
Przycisk „Włącz” po naciśnięciu zmienia etykietę oraz napis na przycisku.

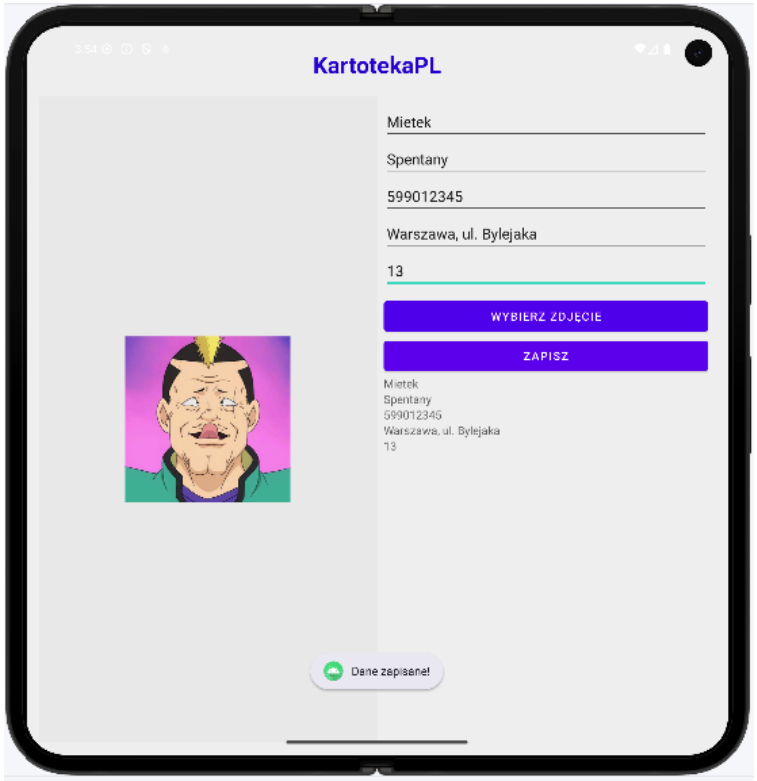


# Zadania do samodzielnego wykonania

## Zadanie 4

Aplikacja w formie formularza. Wykorzystuje dostęp do plików telefonu oraz Shared Preferences do zapisu danych. Dane wprowadzone w formularzu są wyświetlane w liście po przyciskami według poniższego wzoru.



# Zadania do samodzielnego wykonania

**Zadanie 5**

Zadanie polega na utworzeniu aplikacji kalkulatora według wzoru, który zapisuje wszystkie działania w postaci listy w taki sposób, że ostatnie wprowadzone działanie jest pierwszą pozycją na liście.

W projekcie wykorzystano Adapter i kilka styli dla różnych elementów.

# Zadania do samodzielnego wykonania

## Zadanie 6

Zadanie w formie prostej galerii zdjęć. Konieczne będzie dopisanie uprawnień do odczytu z pamięci telefonu oraz umieszczenie w nim zdjęć.

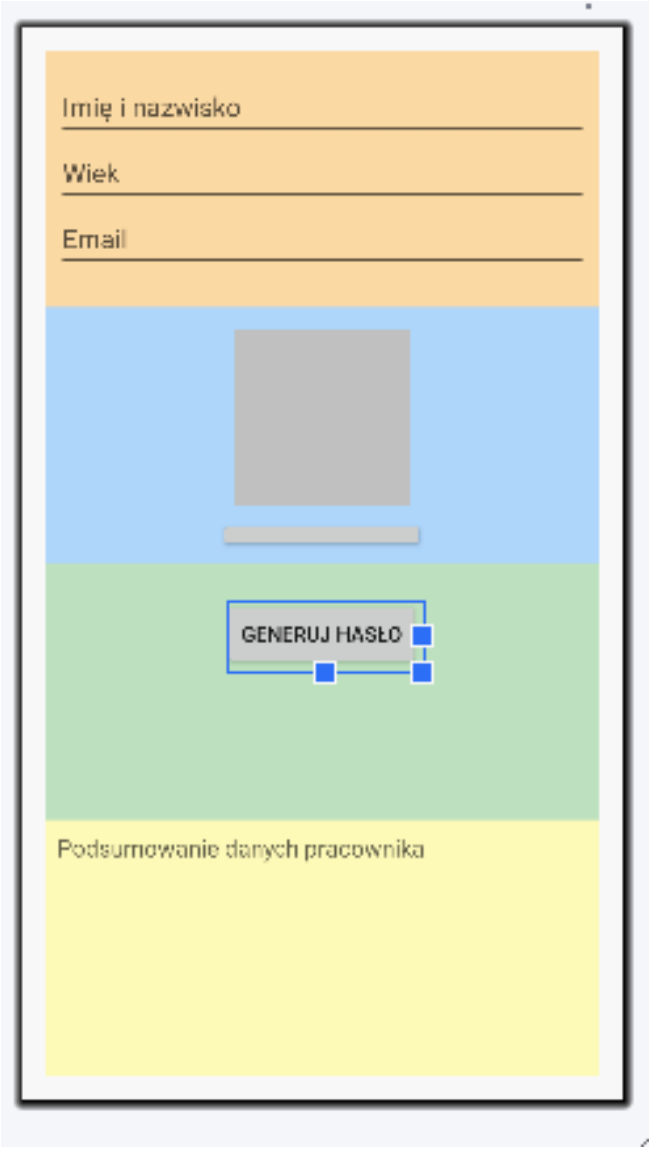
Jak to zrobić w emulatorze?

Należy skopiować pliki do emulatora (po prostu przeciągnąć) a następnie skorzystać z właściwości tablicy i pętli.

# Zadania do samodzielnego wykonania

## Zadanie 7

Przed Tobą aplikacja w formie quizu. Użytkownik odpowiada na pytania naciskając przyciski. Po każdej prawidłowej odpowiedzi wynik jest powiększany o jeden punkt. Potrzebujesz minimum 3 pytań. Do aplikacji dodaj estetyczne tło (kolor lub gradient).




# Zadania do samodzielnego wykonania

## Zadanie 8

Na podstawie przedstawionego layout napisz kod aplikacji, która służy do przygotowania plakietki dla pracownika. Użytkownik wpisuje dane tekstowe, następnie dodaje zdjęcie i stanowisko. Przycisk generuje hasło składające się z 24 znaków. Na ostatniej części layout zostają wypisane wszystkie dane tekstowe wprowadzone w aplikacji.


# Zadania do samodzielnego wykonania

## Zadanie 8

Do zadania potrzebujesz zdjęć kompresora, tokarki i frezarki w formacie 320x240.

Użytkownik wybiera rodzaj maszyny oraz tryb pracy. Wybór maszyny połączony jest z wyborem konkretnego zdjęcia. Zdjęcie zostaje wyświetlone. W normalnym trybie kompresor zużywa 2,5 KW, tokarka 3,0 KW a frezarka 4,0 KW. Intensywny tryb pracy podnosi zużycie o 2,5 KW na godzinę. Maksymalny czas pracy maszyny wynosi 24 godziny.

# Zadanie do samodzielnego wykonania

Projekt zawiera podział na kolumny oraz wyrównanie elementów do środka.

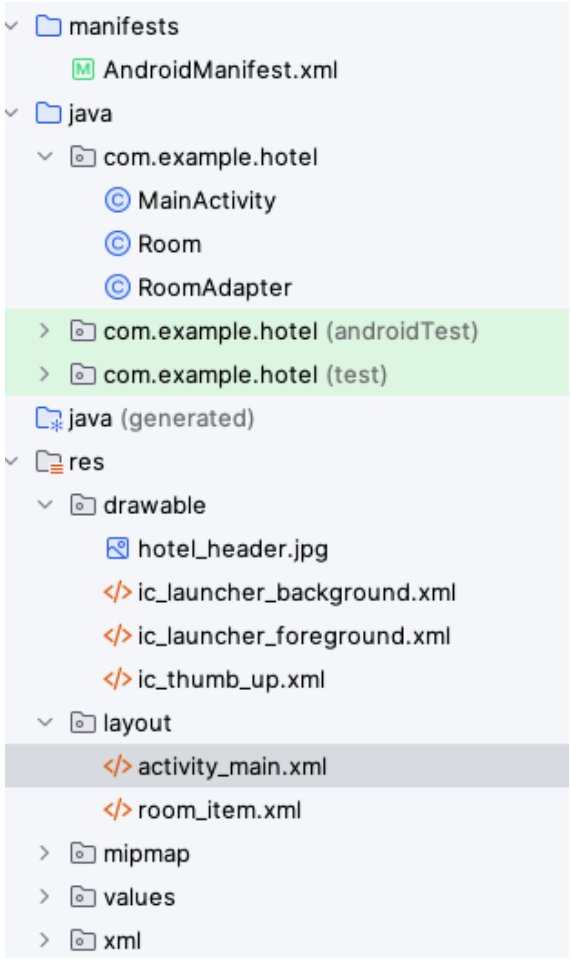
W projekcie zastosowano MaterialUI.

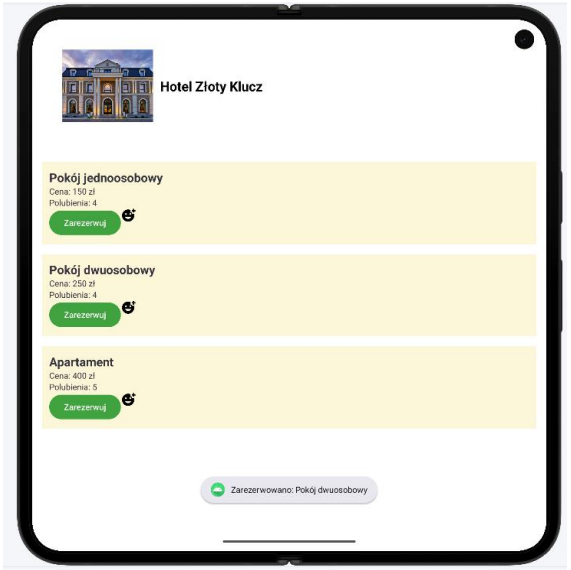

# Zadania do samodzielnego wykonania

**Zadanie 9 (na 5)**

Użytkownik rezerwuje hotel. Po naciśnięciu przycisku wyskakuje komunikat „Zarezerwowano …” . Po każdym naciśnięciu ”buźki” zliczana jest ilość polubień.

W aplikacji korzystano z adaptera.



# Zadania do samodzielnego wykonania

## Zadanie 10 (na 5)

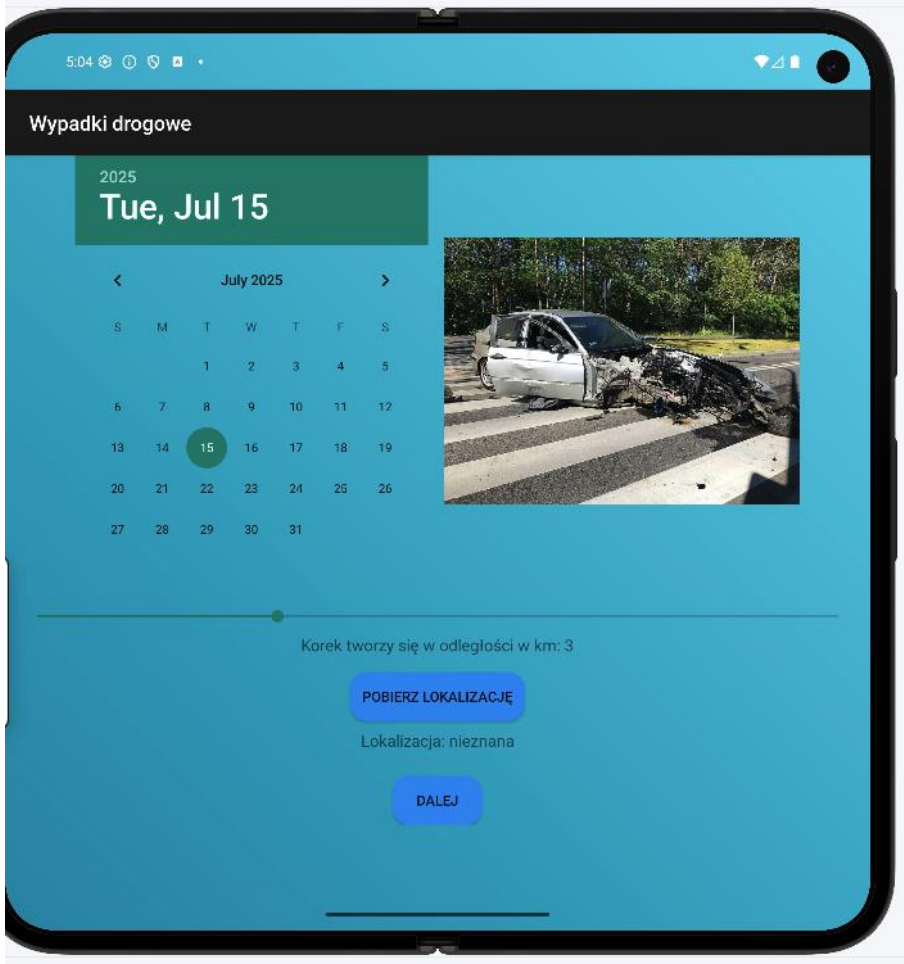
W zadaniu użytkownik wybiera datę a następnie określa w km odległość o tworzącego się zatoru drogowego po wypadku. Pobiera lokalizację dzięki nadanym uprawnieniom.

Po naciśnięciu przycisku ”Dalej” otwiera się druga aktywność z listą zarejestrowanych utrudnień.

Konieczne jest:

- nadanie uprawnień w manifeście;
- dodatkowy XML do przycisków.

Aplikację można rozbudować o dodatkowe możliwości.



# Zadania do samodzielnego wykonania

**Zadanie 12 (na 6)**

Aplikacja w postaci forum użytkowników. Zawiera 3 aktywności:

- Panel logowania z walidacją hasła i loginu;
- Panel, który informuje użytkownika ilu użytkowników jest aktualnie na forum;
- Panel z listą tematów i wypowiedzi (możesz wykorzystać aplikację do listy zadań w tej części).

# Zadania do samodzielnego wykonania

Jak widać, aplikacja zawiera tło w postaci obrazka. W białym fragmencie powinna pojawić się informacja o ilości użytkowników i wpisów. Po naciśnięciu przycisku na drugiej aktywności przenosi do trzeciej aktywności z listą wpisów.

# Zadania do samodzielnego wykonania

## Zadanie 13 (zadanie na 6)

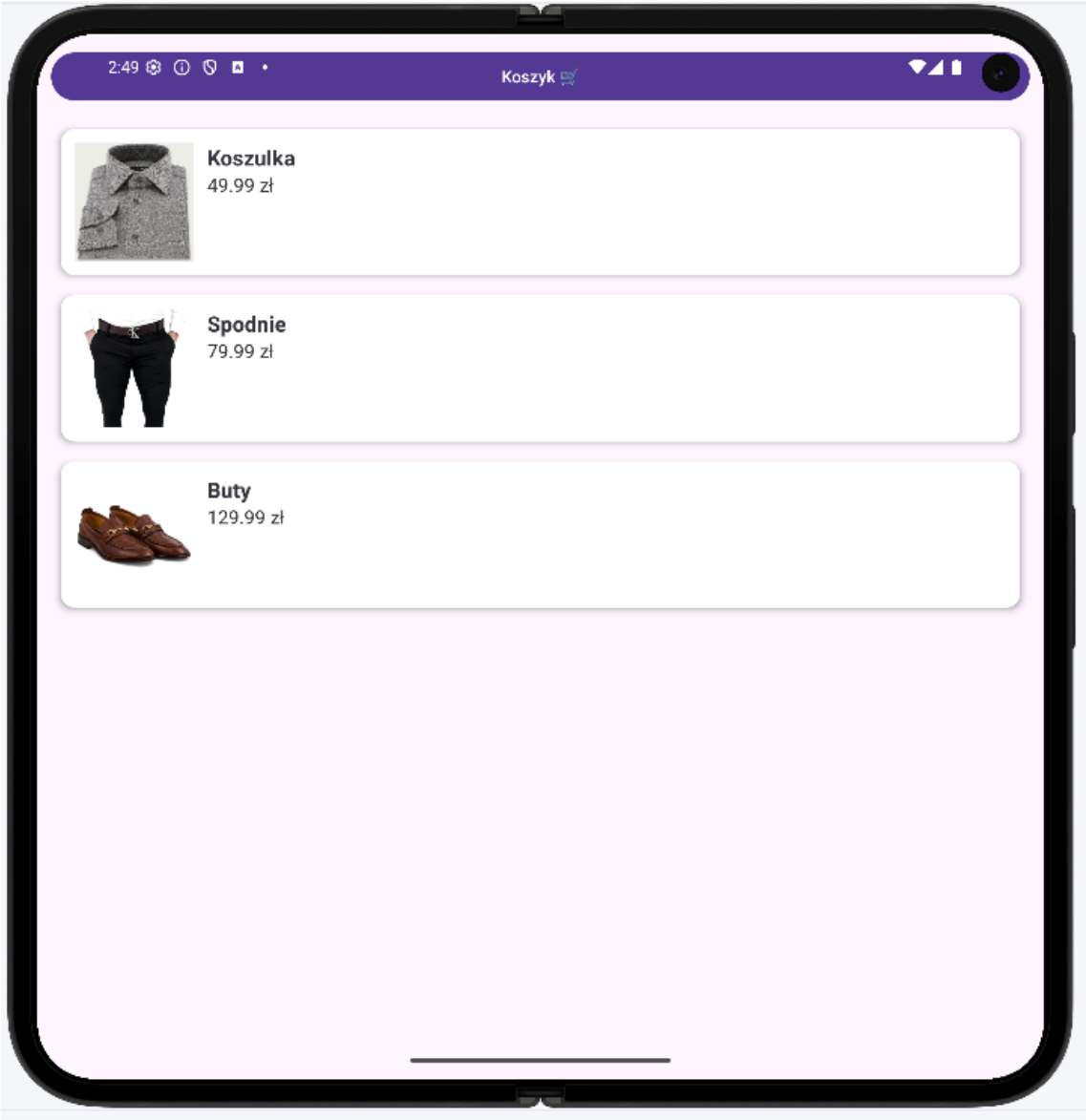
Zadanie w formie prostego sklepu internetowego. Klient wybiera produkty.
Po kliknięciu każdego z nich otwiera się okno z opisem produktu.
Użytkownik może powrócić do pierwszego widoku za pomocą strzałki.

Pod opisem powinien znajdować się przycisk dodania do koszyka.

Co potrzebujesz do wykonania zadania?

Intent, kilka klas JAVA i Adapter.

Rozwiń projekt według własnego pomysłu.



# Zapis i odczyt danych w plikach

Android umożliwia trwały zapis danych w wielu formatach.

Dane mogą być przechowywane w:

- **Plikach (Pamięć aplikacji/Dysk zewnętrzny)**
- **SharedPreferences**
- **Bazach danych SQLite**
- **Danych online (np. Firebase)**

Shared Preferences opiera się na połączeniu klucza i wartości. To sposób prosty, który pozwala na konstrukcję kolekcji danych. Dane kolekcji są przechowywane w plikach JSON w folderze projektu. Czasem konieczne jest dodanie kolejnego DEPENDENCIES (zależności, które łączą strukturę aplikacji z zewnętrznym zasobem).

# Przykład zastosowania

```
SharedPreferences prefs = getSharedPreferences("MojeDane", MODE_PRIVATE);
SharedPreferences.Editor editor = prefs.edit();
editor.putString("nazwa", "Jan");
editor.putInt("wiek", 25);
editor.apply();
```

Zapis

```
SharedPreferences prefs = getSharedPreferences("MojeDane", MODE_PRIVATE);
String nazwa = prefs.getString("nazwa", "Brak");
int wiek = prefs.getInt("wiek", 0);
```

Odczyt

# Zapis i odczyt z pliku zewnętrznego

```
try {
FileOutputStream fos = openFileOutput("plik.txt", MODE_PRIVATE);
fos.write("Witaj Android!".getBytes());
fos.close();
      }
 catch (IOException e) {
 e.printStackTrace();
}
```

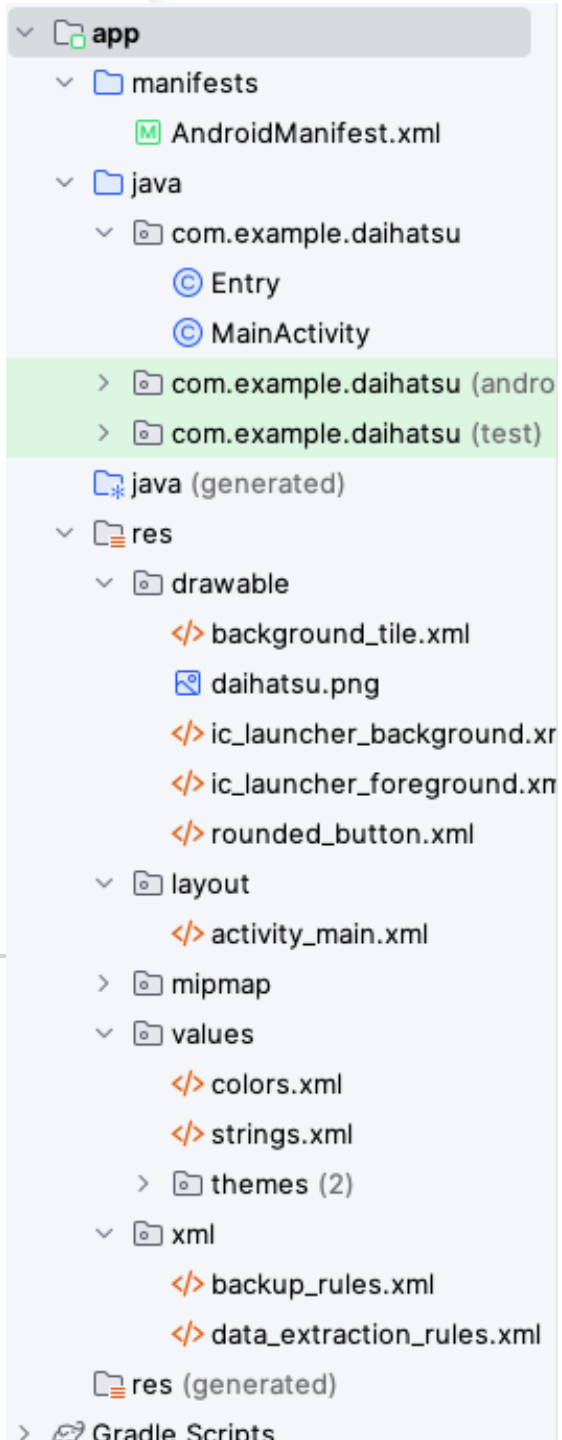

ODCZYT

```
try {
      FileInputStream fis = openFileInput("plik.txt");
      BufferedReader reader = new BufferedReader(new
      InputStreamReader(fis));
      StringBuilder builder = new StringBuilder();
      String linia;
 while ((linia = reader.readLine()) != null)
 {
       builder.append(linia);
 }
      reader.close();
      String wynik = builder.toString();
} catch (IOException e) {
e.printStackTrace();}
```

# Przykładowe zadanie z zapisem do pliku

Użytkownik wprowadza datę, kwotę tankowania, przejechane kilometry i cenę paliwa. Następnie program przelicza dane i wyświetla średnie zużycie paliwa na 100 km.

Poniżej przedstawiono najważniejsze pliki aplikacji.





# Klasy Java (MainActivity)

```
package com.example.daihatsu;

import android.app.DatePickerDialog;
import android.graphics.Color;
import android.graphics.Typeface;
import android.graphics.drawable.Drawable;
import
android.graphics.drawable.GradientDrawable;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
import android.widget.Toast;

import
androidx.appcompat.app.AppCompatActivity;

import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Calendar;
import java.util.Locale;

public class MainActivity extends AppCompatActivity
{

    private EditText editDate, editKm, editLitry,
editCena;
    private TableLayout tableLayout;
    private boolean headerAdded = false;
    private int rowCount = 0; // licznik wierszy

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

        editDate = findViewById(R.id.editDate);
        editKm = findViewById(R.id.editDistance);
        editLitry = findViewById(R.id.editFuel);
        editCena = findViewById(R.id.editPrice);
        tableLayout = findViewById(R.id.tableLayout);
        Button buttonCalculate =
findViewById(R.id.buttonCalculate);
        Button buttonSave =
findViewById(R.id.buttonSave);

        editDate.setOnClickListener(view ->
showDatePicker());

        buttonCalculate.setOnClickListener(v -> {
            String data = editDate.getText().toString();
            String km = editKm.getText().toString();
            String litry = editLitry.getText().toString();
            String cena = editCena.getText().toString();

            if (data.isEmpty() || km.isEmpty() ||
litry.isEmpty() || cena.isEmpty()) {
                Toast.makeText(this, "Uzupełnij wszystkie
pola", Toast.LENGTH_SHORT).show();
                return;
            }

            double kmValue = Double.parseDouble(km);
            double litryValue = Double.parseDouble(litry);
            double cenaValue =
Double.parseDouble(cena);

            double koszt = litryValue * cenaValue;
            double srednieZuzycie = (litryValue / kmValue)
* 100;

            addTableHeader();
            addRowToTable(data, km, litry,
String.format(Locale.US, "%.2f", cenaValue),
                    String.format(Locale.US, "%.2f", koszt),
                    String.format(Locale.US, "%.2f",
srednieZuzycie));

            clearFields();
        });

        buttonSave.setOnClickListener(v ->
saveTableToFile());
    }

    private void showDatePicker() {
        final Calendar c = Calendar.getInstance();
        int year = c.get(Calendar.YEAR);
        int month = c.get(Calendar.MONTH);
        int day = c.get(Calendar.DAY_OF_MONTH);

        DatePickerDialog dpd = new
DatePickerDialog(this,
                (view, year1, month1, dayOfMonth) -> {
                    String date = String.format(Locale.US,
"%04d-%02d-%02d", year1, month1 + 1,
dayOfMonth);
                    editDate.setText(date);
                },
                year, month, day);
        dpd.show();
    }

    private void addTableHeader() {
        if (headerAdded) return;

        TableRow header = new TableRow(this);

header.setBackgroundColor(Color.parseColor("#FFD
D00")); // żółty nagłówek

        header.addView(createHeaderCell("Data"));
        header.addView(createHeaderCell("Km"));
        header.addView(createHeaderCell("Litry"));
        header.addView(createHeaderCell("Cena"));
        header.addView(createHeaderCell("Kwota"));

header.addView(createHeaderCell("Śr./100km"));

        tableLayout.addView(header);
        headerAdded = true;
    }

    private TextView createHeaderCell(String text) {
        TextView tv = new TextView(this);
        tv.setText(text);
        tv.setTextColor(Color.BLACK);
        tv.setTextSize(16);
        tv.setTypeface(Typeface.DEFAULT_BOLD);
        tv.setGravity(Gravity.CENTER);
        tv.setPadding(20, 10, 20, 10);
        tv.setBackground(getCellBorder());
        return tv;
    }

    private TextView createBorderedCell(String text,
boolean isEvenRow) {
        TextView tv = new TextView(this);
        tv.setText(text);
        tv.setTextColor(Color.WHITE);
        tv.setTextSize(14);
        tv.setGravity(Gravity.CENTER);
        tv.setPadding(20, 10, 20, 10);

tv.setBackground(getColoredCellBackground(isEven
Row));
        return tv;
    }

    private Drawable
getColoredCellBackground(boolean isEvenRow) {
        GradientDrawable drawable = new
GradientDrawable();
        drawable.setColor(isEvenRow ?
Color.parseColor("#333333") :
Color.parseColor("#555555")); // naprzemienne
kolory
        drawable.setStroke(2, Color.BLACK);
        return drawable;
    }

    private Drawable getCellBorder() {
        GradientDrawable border = new
GradientDrawable();
        border.setColor(Color.TRANSPARENT);
        border.setStroke(2, Color.BLACK);
        return border;
    }

    private void addRowToTable(String data, String
km, String litry, String cena, String kwota, String
srednieZuzycie) {
        TableRow row = new TableRow(this);

        boolean isEven = rowCount % 2 == 0;
        row.addView(createBorderedCell(data, isEven));
        row.addView(createBorderedCell(km, isEven));
        row.addView(createBorderedCell(litry, isEven));
        row.addView(createBorderedCell(cena, isEven));
        row.addView(createBorderedCell(kwota,
isEven));

row.addView(createBorderedCell(srednieZuzycie,
isEven));
        tableLayout.addView(row);
        rowCount++;
    }

    private void saveTableToFile() {
        StringBuilder data = new StringBuilder();
        int rowCount = tableLayout.getChildCount();

        for (int i = 1; i < rowCount; i++) {
            TableRow row = (TableRow)
tableLayout.getChildAt(i);
            int cellCount = row.getChildCount();

            for (int j = 0; j < cellCount; j++) {
                TextView tv = (TextView) row.getChildAt(j);
                data.append(tv.getText().toString());

                if (j < cellCount - 1) {
                    data.append(" | ");
                }
            }
            data.append("\n");
        }

        try {
            FileOutputStream fos =
openFileOutput("dane_tankowania.txt",
MODE_PRIVATE);
            fos.write(data.toString().getBytes());
            fos.close();
            Toast.makeText(this, "Zapisano do pliku",
Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            Toast.makeText(this, "Błąd zapisu",
Toast.LENGTH_SHORT).show();
            e.printStackTrace();
        }
    }

    private void clearFields() {
        editKm.setText("");
        editLitry.setText("");
        editCena.setText("");
    }
}
```

# Klasy JAVA (Entry)

```
package com.example.daihatsu;
public class Entry {
    private double distance;
    private double fuel;
    private double price;
    private double consumption;
    private double cost;

    public Entry(double distance, double fuel, double price, double consumption, double cost){
        this.distance = distance;
        this.fuel = fuel;
        this.price = price;
        this.consumption = consumption;
        this.cost = cost;
    }

    // Gettery
    public double getDistance() { return distance; }
    public double getFuel(){ return fuel; }
    public double getPrice(){ return price; }
    public double getConsumption() { return consumption; }
    public double getCost() { return cost; }

    @Override
    public String toString() {
        return String.format("Km: %.0f, L: %.2f, zł/l: %.2f, L/100km: %.2f, zł/100km: %.2f",
                distance, fuel, price, consumption, cost);
    }
}
```

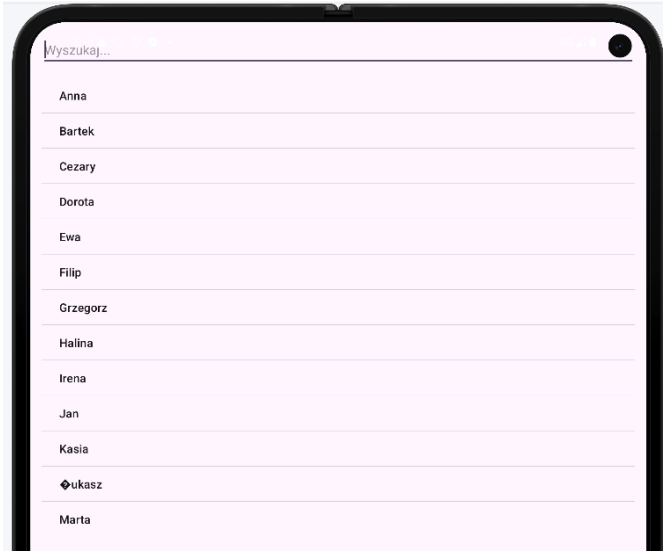
# XML do stylizacji

```xml
<?xml version="1.0" encoding="utf-8"?>
<bitmap
xmlns:android="http://schemas.android
.com/apk/res/android"
   android:src="@drawable/daihatsu"
   android:tileMode="repeat" />
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<shape
xmlns:android="http://schemas.androi
d.com/apk/res/android"
android:shape="rectangle">
   <solid android:color="#FF5722"/>
<!-- kolor tła przycisku -->
   <corners android:radius="25dp"/>
<!-- promień zaokrąglenia -->
   <padding android:left="10dp"
android:top="10dp"
android:right="10dp"
android:bottom="10dp"/>
</shape>
```

```xml
?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/
android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:backgroundTint="#FF5722">

  <!-- Tło jako ImageView -->
  <ImageView
    android:id="@+id/backgroundImage"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:scaleType="centerCrop"
    android:src="@drawable/daihatsu"/>

  <!-- Twoje elementy UI na wierzchu -->
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="100dp"
    android:orientation="vertical"
    android:padding="16dp">

    <EditText
      android:id="@+id/editDate"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Wybierz datę"
      android:textColor="@color/yellow"
      android:textSize="28dp"
      android:focusable="false"
      android:clickable="true"
      android:inputType="none" />

    <EditText
      android:id="@+id/editAmount"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Kwota tankowania (zł)"
      android:textColor="@color/yellow"
      android:textSize="24dp"
      android:inputType="numberDecimal" />

    <EditText
      android:id="@+id/editDistance"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="@string/przejechane_km"
      android:textColor="@color/yellow"
      android:textSize="24dp"
      android:inputType="numberDecimal" />

    <EditText
      android:id="@+id/editFuel"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="@string/zuzyte_paliwo_litry"
      android:textColor="@color/yellow"
      android:textSize="24dp"
      android:inputType="numberDecimal" />

    <EditText
      android:id="@+id/editPrice"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="@string/cena_paliwa_za_litr"
      android:textColor="@color/yellow"
      android:textSize="24dp"
      android:inputType="numberDecimal" />

    <Button
      android:id="@+id/buttonCalculate"

android:background="@drawable/rounded_button"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="16dp"

android:text="@string/oblicz_zuzycie_i_koszt_100_km"
/>

    <TextView
      android:id="@+id/textConsumption"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="24dp"
      android:text=""
      android:textSize="18sp" />

    <TextView
      android:id="@+id/textCost"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="8dp"
      android:text=""
      android:textSize="24sp"
      android:textColor="@color/yellow"/>

    <ScrollView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="16dp">

      <TableLayout
        android:id="@+id/tableLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:stretchColumns="*"
        android:padding="8dp"
        android:background="#66000000">

        <!-- Nagłówki tabeli -->
        <TableRow>
          <TextView android:text="Data"
android:textColor="#FFFF00"
android:textStyle="bold"/>
          <TextView android:text="Km"
android:textColor="#FFFF00"
android:textStyle="bold"/>
          <TextView android:text="Litry"
android:textColor="#FFFF00"
android:textStyle="bold"/>
          <TextView android:text="Cena/l"
android:textColor="#FFFF00"
android:textStyle="bold"/>
          <TextView android:text="Kwota"
android:textColor="#FFFF00"
android:textStyle="bold"/>
        </TableRow>

      </TableLayout>
    </ScrollView>

    <Button
      android:id="@+id/buttonSave"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="16dp"
      android:text="Zapisz do pliku" />

  </LinearLayout>

</FrameLayout>
```

# Wyszukiwarka danych z pliku + wyświetlenie



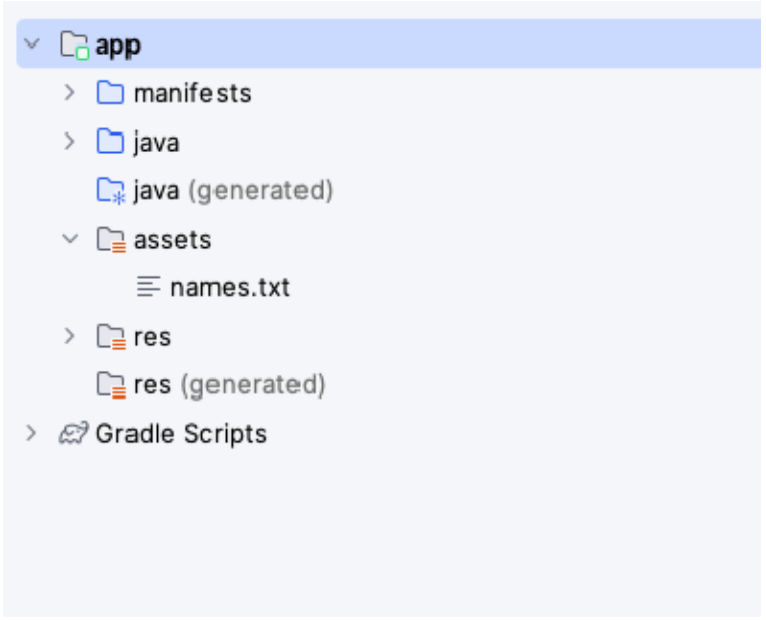


Prosta wyszukiwarka danych z pliku a w nim lista z imionami.

Uwaga na kodowanie – plik z danymi musi mieć odpowiedni UTF (patrz: Łukasz bez polskiej litery).

Dane z pliku są pobrane w całości a następnie po rozpoczęciu wpisywania lista zmniejsza się do rozmiaru odpowiadającemu zgodności wpisanych danych z danymi umieszczonymi w pliku.

# Kod wyszukiwarki




```
package com.example.searchapp;

import android.content.res.AssetManager;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;

import androidx.appcompat.app.AppCompatActivity;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

  EditText searchEditText;
  ListView listView;

  ArrayAdapter<String> adapter;
  ArrayList<String> dataList;

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

    searchEditText = findViewById(R.id.searchEditText);
    listView = findViewById(R.id.listView);

    dataList = new ArrayList<>();

    // Wczytaj dane z pliku w assets
    readDataFromAssets();

    adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, dataList);
    listView.setAdapter(adapter);

    searchEditText.addTextChangedListener(new TextWatcher() {
      @Override
      public void beforeTextChanged(CharSequence s, int start, int
count, int after) { }

      @Override
      public void onTextChanged(CharSequence s, int start, int
before, int count) {
        adapter.getFilter().filter(s);
      }

      @Override
      public void afterTextChanged(Editable s) { }
    });
  }

  private void readDataFromAssets() {
    AssetManager assetManager = getAssets();
    try {
      InputStream inputStream = assetManager.open("names.txt");
      BufferedReader reader = new BufferedReader(new
InputStreamReader(inputStream));
      String line;
      while ((line = reader.readLine()) != null) {
        dataList.add(line);
      }
      reader.close();
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

# Proponowany Layout

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

xmlns:android="http://schemas.android.com/apk/res/
android"
  xmlns:app="http://schemas.android.com/apk/res-
auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"

android:background="@color/material_dynamic_secon
dary90"
  android:padding="16dp">

  <!-- Pole wyszukiwania z ikoną -->
  <androidx.appcompat.widget.LinearLayoutCompat
    android:layout_width="match_parent"
    android:layout_height="48dp"

android:background="@drawable/search_background"
    android:paddingStart="12dp"
    android:paddingEnd="12dp"
    android:gravity="center_vertical"
    android:elevation="2dp">

    <ImageView
      android:layout_width="24dp"
      android:layout_height="24dp"

android:src="@android:drawable/ic_menu_search"
      app:tint="#616161"/>

    <EditText
      android:id="@+id/searchEditText"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"

android:background="@android:color/transparent"
      android:hint="@string/wyszukaj"
      android:textColor="#212121"
      android:textColorHint="#757575"
      android:padding="12dp"
      android:textSize="16sp"
      android:inputType="text"/>

  </androidx.appcompat.widget.LinearLayoutCompat>

  <!-- Lista z podkreśleniami -->
  <ListView
    android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:divider="#CCCCCC"
    android:dividerHeight="1dp"
    android:layout_marginTop="16dp"
    android:background="#FFFFFF"
    android:paddingTop="4dp"
    android:paddingBottom="4dp"
    android:clipToPadding="false"/>

</LinearLayout>
```

# Wprowadź – Wyświetl - Wyszukaj


Aplikacja tworzy plik z listą imion, wyświetla aktualną listę, pozwala wyszukać oraz dodawać kolejne. Zastosowano również MaterialUI dla szybkiej zmiany tła.

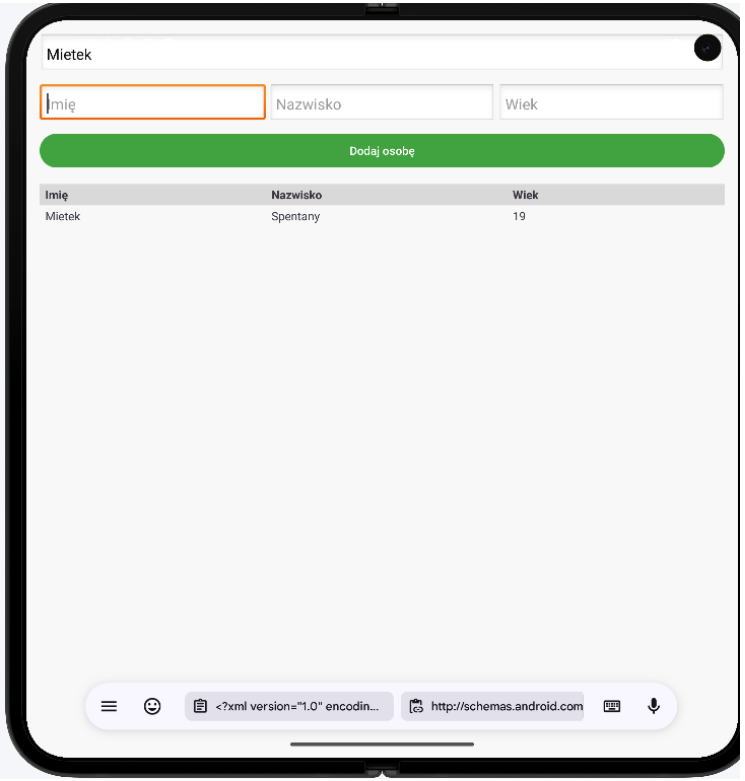
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

xmlns:android="http://schemas.android.com/apk/res
/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"

android:background="@color/material_dynamic_pri
mary40"
  android:padding="16dp">

  <!-- Pole wyszukiwania -->
  <EditText
    android:id="@+id/searchEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Szukaj..."
    android:inputType="text"

android:background="@android:drawable/edit_text"
    android:padding="10dp"
    android:textSize="16sp"
    android:layout_marginBottom="12dp"/>

  <!-- Pole dodawania -->
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <EditText
      android:id="@+id/addEditText"
      android:layout_width="0dp"
      android:layout_weight="1"
      android:layout_height="wrap_content"
      android:hint="Dodaj nowy wpis"
      android:inputType="text"

android:background="@android:drawable/edit_text"
      android:padding="10dp"
      android:textSize="16sp"/>

    <Button
      android:id="@+id/addButton"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Dodaj"
      android:layout_marginStart="8dp"
      android:backgroundTint="@color/black"/>
  </LinearLayout>

  <!-- Lista -->
  <ListView
    android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:divider="#CCCCCC"
    android:dividerHeight="1dp"
    android:layout_marginTop="16dp"
    android:background="#FFFFFF"
    android:clipToPadding="false"/>
</LinearLayout>
```

# Wprowadź-Wyświetl-Wyszukaj

```
package com.example.przelotka;

import android.os.Bundle;
import android.widget.*;
import androidx.appcompat.app.AppCompatActivity;
import java.io.*;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    EditText searchEditText, addEditText;
    Button addButton;
    ListView listView;
    ArrayList<String> itemsList = new ArrayList<>();
    ArrayAdapter<String> adapter;
    File dataFile;

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

        searchEditText = findViewById(R.id.searchEditText);
        addEditText = findViewById(R.id.addEditText);
        addButton = findViewById(R.id.addButton);
        listView = findViewById(R.id.listView);

        dataFile = new File(getFilesDir(), "names.txt");
        ensureFileExists();
        loadDataFromFile();

        adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, new ArrayList<>(itemsList));
        listView.setAdapter(adapter);

        searchEditText.addTextChangedListener(new
android.text.TextWatcher() {
            @Override public void beforeTextChanged(CharSequence s, int
start, int count, int after) {}
            @Override public void onTextChanged(CharSequence s, int
start, int before, int count) {
                adapter.getFilter().filter(s);
            }
            @Override public void afterTextChanged(android.text.Editable
s) {}
        });

        addButton.setOnClickListener(v -> {
            String newItem = addEditText.getText().toString().trim();
            if (!newItem.isEmpty()) {
                itemsList.add(newItem);
                writeDataToFile();
                adapter.clear();
                adapter.addAll(itemsList);
                adapter.notifyDataSetChanged();
                addEditText.setText("");
                Toast.makeText(this, "Dodano!",
Toast.LENGTH_SHORT).show();
            }
        });
    }

    void ensureFileExists() {
        if (!dataFile.exists()) {
            try (BufferedWriter writer = new BufferedWriter(new
FileWriter(dataFile))) {
                writer.write("Anna\nBartek\nCezary\nDorota\nEwa");
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    void loadDataFromFile() {
        itemsList.clear();
        try (BufferedReader reader = new BufferedReader(new
FileReader(dataFile))) {
            String line;
            while ((line = reader.readLine()) != null) {
                itemsList.add(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    void writeDataToFile() {
        try (BufferedWriter writer = new BufferedWriter(new
FileWriter(dataFile))) {
            for (String item : itemsList) {
                writer.write(item);
                writer.newLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

# www
# z tabelą

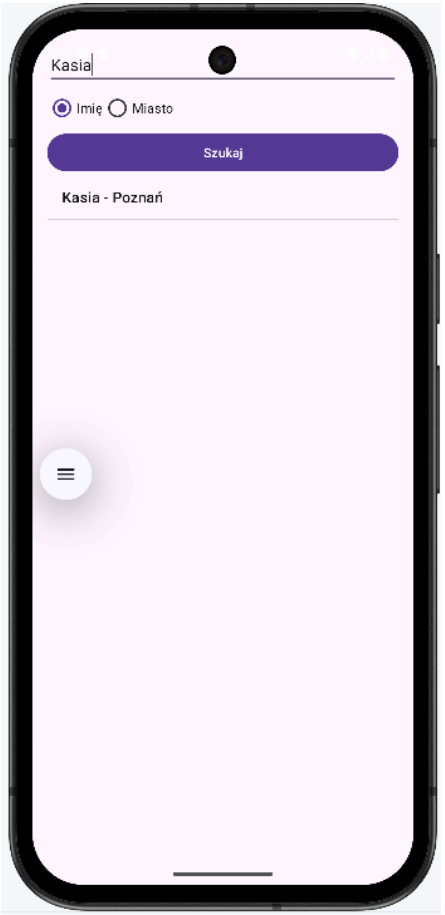
```
•	package com.example.tableapp;

import androidx.appcompat.app.AppCompatActivity;

import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.*;

import java.io.*;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private EditText searchEditText, inputFirstName, inputLastName, inputAge;
    private Button addButton;
    private TableLayout tableLayout;

    private List<String[]> itemList = new ArrayList<>();
    private final String FILE_NAME = "data.txt";

    @Override
    protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

        searchEditText = findViewById(R.id.searchEditText);
        inputFirstName = findViewById(R.id.inputFirstName);
        inputLastName = findViewById(R.id.inputLastName);
        inputAge = findViewById(R.id.inputAge);
        addButton = findViewById(R.id.addButton);
        tableLayout = findViewById(R.id.tableLayout);

        loadData();
        updateTable(itemList);

        addButton.setOnClickListener(v -> {
            String first = inputFirstName.getText().toString().trim();
            String last = inputLastName.getText().toString().trim();
            String age = inputAge.getText().toString().trim();
            if (!first.isEmpty() && !last.isEmpty() && !age.isEmpty()) {
                itemList.add(new String[]{first, last, age});
                saveData();
                clearInputs();

filterData(searchEditText.getText().toString());
            }
        });


searchEditText.addTextChangedListener(new android.text.TextWatcher() {
            @Override public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
            @Override public void onTextChanged(CharSequence s, int start, int before, int count) {
                filterData(s.toString());
            }
            @Override public void afterTextChanged(android.text.Editable s) {}
        });
    }

    private void clearInputs() {
        inputFirstName.setText("");
        inputLastName.setText("");
        inputAge.setText("");
    }

    private void loadData() {
        itemList.clear();
        File file = new File(getFilesDir(), FILE_NAME);
        if (file.exists()) {
            try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
                String line;
                while ((line = reader.readLine()) != null) {
                    String[] parts = line.split(",");
                    if (parts.length == 3) {
                        itemList.add(parts);
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    private void saveData() {
        File file = new File(getFilesDir(), FILE_NAME);
        try (BufferedWriter writer = new BufferedWriter(new FileWriter(file))) {
            for (String[] item : itemList) {
                writer.write(String.join(",", item));
                writer.newLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void filterData(String query) {
        List<String[]> filtered = new ArrayList<>();
        for (String[] item : itemList) {
            if ((item[0] + " " + item[1] + " " + item[2]).toLowerCase().contains(query.toLowerCase())) {
                filtered.add(item);
            }
        }
        updateTable(filtered);
    }

    private void updateTable(List<String[]> data) {
        tableLayout.removeAllViews();

        TableRow header = new TableRow(this);

header.setBackgroundColor(0xFFE0E0E0);
        String[] headers = {"Imię", "Nazwisko", "Wiek"};

        for (String h : headers) {
            TextView tv = new TextView(this);
            tv.setText(h);
            tv.setTypeface(null, Typeface.BOLD);
            tv.setPadding(16, 8, 16, 8);
            header.addView(tv);
        }
        tableLayout.addView(header);

        for (String[] rowItem : data) {
            TableRow row = new TableRow(this);
            for (String cellText : rowItem) {
                TextView cell = new TextView(this);
                cell.setText(cellText);
                cell.setPadding(16, 8, 16, 8);
                row.addView(cell);
            }
            tableLayout.addView(row);
        }
    }
}
```




# WWW z tabelą

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="16dp"
  android:background="#FAFAFA">

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <!-- Wyszukiwanie -->
    <EditText
      android:id="@+id/searchEditText"

android:layout_width="match_parent"

android:layout_height="wrap_content"
      android:hint="Wyszukaj..."
      android:inputType="text"
      android:padding="8dp"

android:background="@android:drawable
/edit_text"
      android:textColor="#000000"
      android:textColorHint="#888888" />

    <!-- Pola dodawania danych -->
    <LinearLayout

android:layout_width="match_parent"

android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:layout_marginTop="12dp"
      android:weightSum="3">

      <EditText
        android:id="@+id/inputFirstName"
        android:layout_width="0dp"

android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="Imię"
        android:inputType="text"
        android:padding="8dp"

android:background="@android:drawable
/edit_text"
        android:textColor="#000000"/>

      <EditText
        android:id="@+id/inputLastName"
        android:layout_width="0dp"

android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="Nazwisko"
        android:inputType="text"
        android:padding="8dp"
        android:layout_marginStart="4dp"

android:background="@android:drawable
/edit_text"
        android:textColor="#000000"/>

      <EditText
        android:id="@+id/inputAge"
        android:layout_width="0dp"

android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="Wiek"
        android:inputType="number"
        android:padding="8dp"
        android:layout_marginStart="4dp"

android:background="@android:drawable
/edit_text"
        android:textColor="#000000"/>
    </LinearLayout>

    <!-- Przycisk -->
    <Button
      android:id="@+id/addButton"

android:layout_width="match_parent"

android:layout_height="wrap_content"
      android:text="Dodaj osobę"
      android:textColor="#FFFFFF"
      android:backgroundTint="#4CAF50"
      android:layout_marginTop="8dp"/>

    <!-- Tabela -->
    <TableLayout
      android:id="@+id/tableLayout"

android:layout_width="match_parent"

android:layout_height="wrap_content"
      android:stretchColumns="*"
      android:layout_marginTop="16dp"
/>

  </LinearLayout>
</ScrollView>
```

# Wyszukiwarka z RadioButton




```
package com.example.radiowyszukiwarka;

import android.os.Bundle;
import android.widget.*;
import
androidx.appcompat.app.AppCompatActivity
;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends
AppCompatActivity {

  EditText editTextSearch;
  RadioGroup radioGroup;
  RadioButton radioName, radioCity;
  Button buttonSearch;
  ListView listView;

  List<User> userList = new ArrayList<>();
  ArrayAdapter<User> adapter;
  List<User> filteredList = new ArrayList<>();

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

    editTextSearch =
findViewById(R.id.editTextSearch);
    radioGroup =
findViewById(R.id.radioGroup);
    radioName =
findViewById(R.id.radioName);
    radioCity = findViewById(R.id.radioCity);
    buttonSearch =
findViewById(R.id.buttonSearch);
    listView = findViewById(R.id.listView);

    // Przykładowe dane
    userList.add(new User("Anna",
"Warszawa"));
    userList.add(new User("Piotr", "Kraków"));
    userList.add(new User("Kasia",
"Poznań"));
    userList.add(new User("Marek",
"Gdańsk"));
    userList.add(new User("Paweł",
"Warszawa"));

    adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1,
filteredList);
    listView.setAdapter(adapter);

    buttonSearch.setOnClickListener(v ->
searchUsers());
  }

  private void searchUsers(){
    String query =
editTextSearch.getText().toString().trim().toLo
werCase();
    filteredList.clear();

    if (query.isEmpty()){
      Toast.makeText(this, "Wpisz frazę do
wyszukania", Toast.LENGTH_SHORT).show();
      return;
  Kasia    }

    boolean searchByName =
radioName.isChecked();

    for (User user : userList) {
      if (searchByName &&
user.getName().toLowerCase().contains(quer
y)){
        filteredList.add(user);
      } else if (!searchByName &&
user.getCity().toLowerCase().contains(query))
{
        filteredList.add(user);
      }
    }

    if (filteredList.isEmpty()){
      Toast.makeText(this, "Brak wyników",
Toast.LENGTH_SHORT).show();
    }

    adapter.notifyDataSetChanged();
  }
}
```

# Odczyt i zapis do SQLite

```
try {
    FileOutputStream fos = openFileOutput("plik.txt", MODE_PRIVATE);
    fos.write("Witaj Android!".getBytes());
    fos.close();}
catch (IOException e) {
   e.printStackTrace();
}
```

```
try {
    FileInputStream fis = openFileInput("plik.txt");
    BufferedReader reader = new BufferedReader(new
    InputStreamReader(fis));
    StringBuilder builder = new StringBuilder();
    String linia;
    while ((linia = reader.readLine()) != null) {
        builder.append(linia);
 }
reader.close();
String wynik = builder.toString();
}
catch (IOException e) {
    e.printStackTrace();
}
```

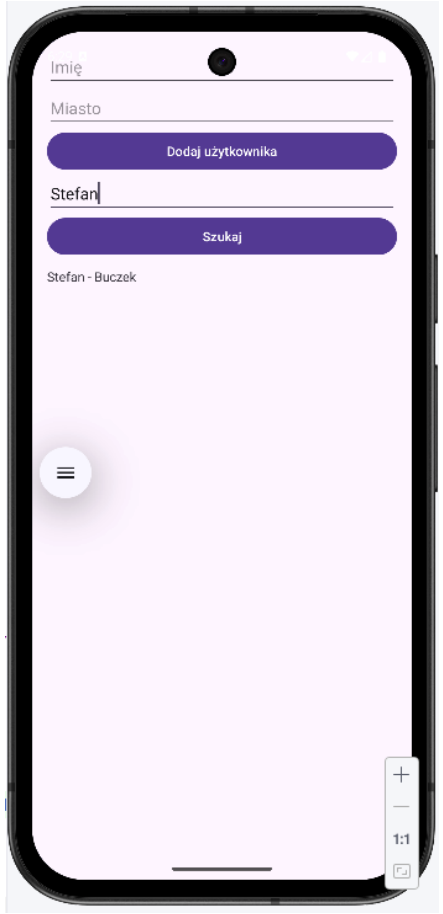
# Odczyt i zapis do SQLite

## Tworzenie bazy danych:

Klasa za pomocą dwóch metod tworzy tabelę oraz usuwa, jeżeli mamy już utworzoną pod taką nazwą.

```
public class DBHelper extends SQLiteOpenHelper {
      public DBHelper(Context context) {
      super(context, "Uzytkownicy.db", null, 1);
      }

@Override
public void onCreate(SQLiteDatabase db) {
     db.execSQL("CREATE TABLE uzytkownik(id INTEGER
PRIMARY KEY, imie TEXT, wiek INTEGER)");
     }

@Override
 public void onUpgrade(SQLiteDatabase db, int oldV, int
newV) {
     db.execSQL("DROP TABLE IF EXISTS uzytkownik");
     onCreate(db);
 }
}
```

# Przykładowa aplikacja z zapisem do bazy i wyszukiwarką



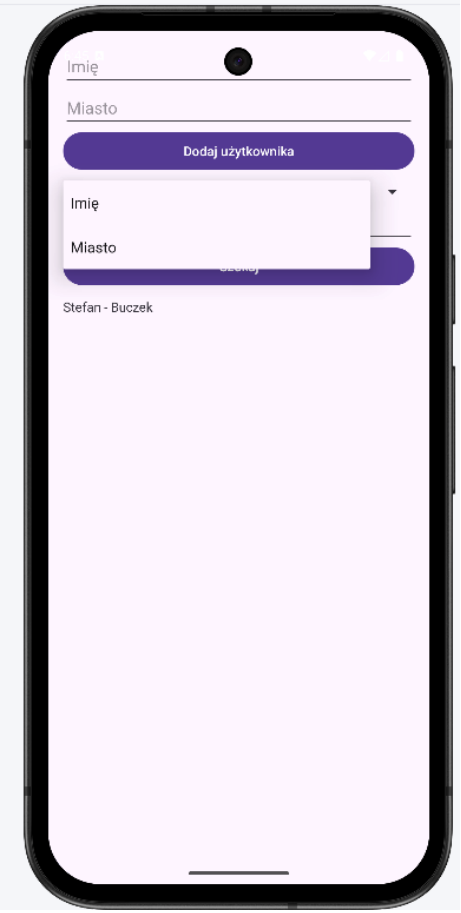

```
package com.example.apkabaza;

import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.*;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    EditText editName, editCity, editSearch;
    Button buttonAdd, buttonSearch;
    TextView textResults;

    DatabaseHelper db;

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

        editName = findViewById(R.id.editName);
        editCity = findViewById(R.id.editCity);
        editSearch = findViewById(R.id.editSearch);
        buttonAdd = findViewById(R.id.buttonAdd);
        buttonSearch = findViewById(R.id.buttonSearch);
        textResults = findViewById(R.id.textResults);

        db = new DatabaseHelper(this);

        buttonAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = editName.getText().toString().trim();
                String city = editCity.getText().toString().trim();
                if (!name.isEmpty() && !city.isEmpty()) {
                    boolean inserted = db.addUser(name, city);
                    Toast.makeText(MainActivity.this, inserted ? "Dodano!" :
"Błąd!", Toast.LENGTH_SHORT).show();
                    editName.setText("");
                    editCity.setText("");
                } else {
                    Toast.makeText(MainActivity.this, "Uzupełnij wszystkie pola!",
Toast.LENGTH_SHORT).show();
                }
            }
        });

        buttonSearch.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String query = editSearch.getText().toString().trim();
                Cursor cursor = db.searchUsersByName(query);
                StringBuilder builder = new StringBuilder();
                if (cursor.getCount() == 0) {
                    builder.append("Brak wyników.");
                } else {
                    while (cursor.moveToNext()) {
                        String name = cursor.getString(1);
                        String city = cursor.getString(2);
                        builder.append(name).append(" -
").append(city).append("\n");
                    }
                }
                textResults.setText(builder.toString());
                cursor.close();
            }
        });
    }
}
```

# Przykładowa aplikacja z zapisem do bazy i wyszukiwarką

```
package com.example.apkabaza;


import android.database.Cursor;

import android.os.Bundle;

import android.view.View;

import android.widget.*;

import androidx.appcompat.app.AppCompatActivity;


public class MainActivity extends AppCompatActivity {


  EditText editName, editCity, editSearch;

  Button buttonAdd, buttonSearch;

  TextView textResults;


  DatabaseHelper db;


  @Override

  protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);


    editName = findViewById(R.id.editName);

    editCity = findViewById(R.id.editCity);

    editSearch = findViewById(R.id.editSearch);

    buttonAdd = findViewById(R.id.buttonAdd);

    buttonSearch = findViewById(R.id.buttonSearch);

    textResults = findViewById(R.id.textResults);


    db = new DatabaseHelper(this);


    buttonAdd.setOnClickListener(new
View.OnClickListener(){

      @Override

      public void onClick(View v){

        String name = editName.getText().toString().trim();

        String city = editCity.getText().toString().trim();

        if (!name.isEmpty() && !city.isEmpty()) {

          boolean inserted = db.addUser(name, city);

          Toast.makeText(MainActivity.this, inserted ?
"Dodano!" : "Błąd!", Toast.LENGTH_SHORT).show();

          editName.setText("");

          editCity.setText("");

        } else {

          Toast.makeText(MainActivity.this, "Uzupełnij
wszystkie pola!", Toast.LENGTH_SHORT).show();

        }

      }

    });


    buttonSearch.setOnClickListener(new
View.OnClickListener(){

      @Override

      public void onClick(View v){

        String query = editSearch.getText().toString().trim();

        Cursor cursor = db.searchUsersByName(query);

        StringBuilder builder = new StringBuilder();

        if (cursor.getCount() == 0) {

          builder.append("Brak wyników.");

        } else {

          while (cursor.moveToNext()) {

            String name = cursor.getString(1);

            String city = cursor.getString(2);

            builder.append(name).append(" -
").append(city).append("\n");

          }

        }

        textResults.setText(builder.toString());

        cursor.close();

      }

    });

  }

}
```

**Klasa pomocnicza, która zawiera logikę wyszukiwania. Znajduje się w niej mechanizm wyszukiwania oraz odwołania do zasobu ukrytego w bazie.**

# Rozszerzenie aplikacji o wyszukiwanie za pomocą Spinnera




```
public Cursor searchUsers(String column, String value) {
    SQLiteDatabase db = this.getReadableDatabase();
    return db.rawQuery("SELECT * FROM " + TABLE_USERS + " WHERE " + column + " LIKE ?", new String[]{"%" + value + "%"});
}
```

Dodatkowa metoda w klasie Helper

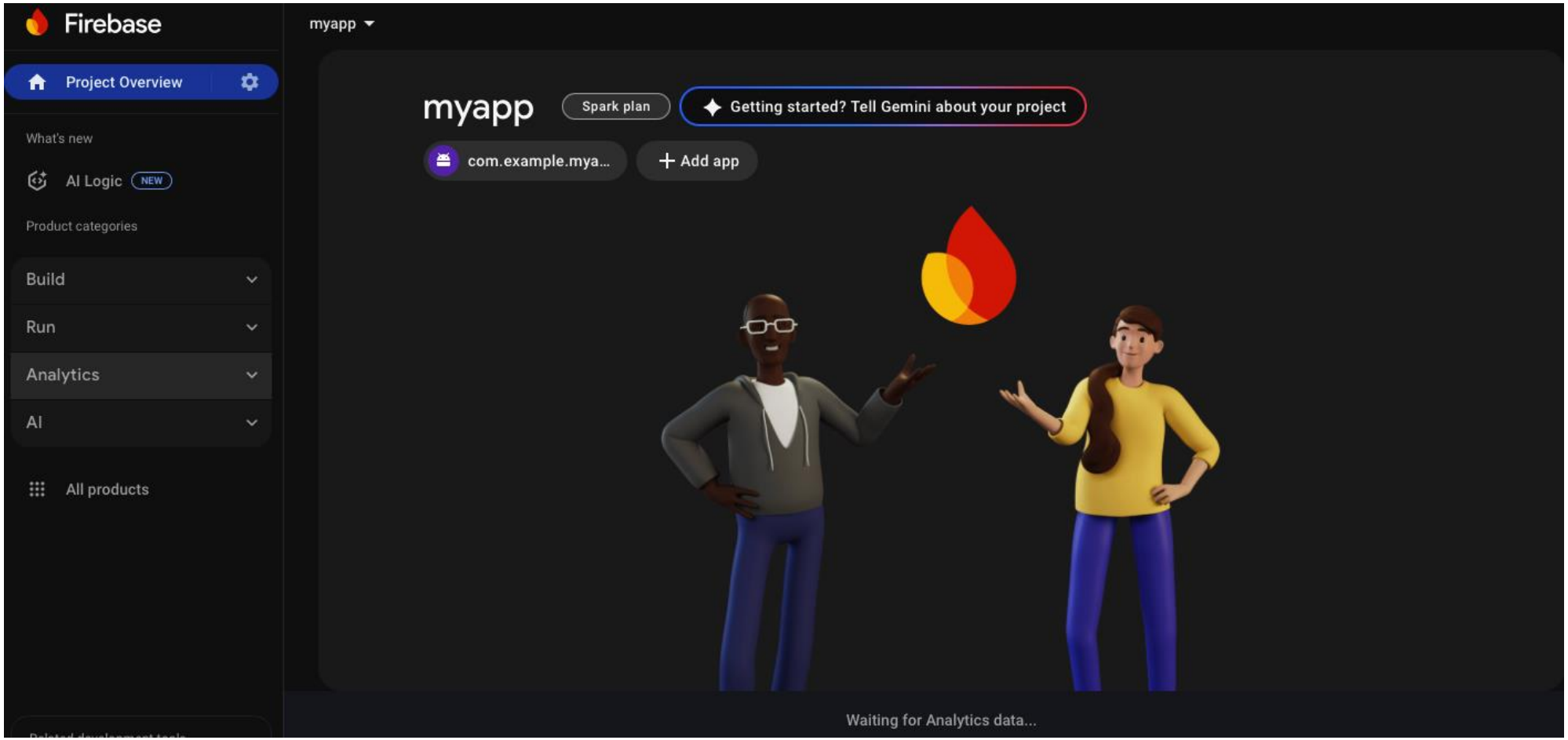
Dodatkowy Spinner

```
<Spinner
    android:id="@+id/spinnerSearchBy"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"/>
```

# Rozszerzenie aplikacji o wyszukiwanie za pomocą Spinnera

```
package com.example.apkabaza;

import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.*;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    EditText editName, editCity, editSearch;
    Button buttonAdd, buttonSearch;
    Spinner spinnerSearchBy;
    TextView textResults;

    DatabaseHelper db;

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

        editName = findViewById(R.id.editName);
        editCity = findViewById(R.id.editCity);
        editSearch = findViewById(R.id.editSearch);
        buttonAdd = findViewById(R.id.buttonAdd);
        buttonSearch = findViewById(R.id.buttonSearch);
        spinnerSearchBy = findViewById(R.id.spinnerSearchBy);
        textResults = findViewById(R.id.textResults);

        db = new DatabaseHelper(this);

        // 🔽 Ustawienie opcji Spinnera
        ArrayAdapter<String> spinnerAdapter = new ArrayAdapter<>(this,
                android.R.layout.simple_spinner_item,
                new String[]{"Imię", "Miasto"});

spinnerAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinnerSearchBy.setAdapter(spinnerAdapter);

        // 🔘 Przycisk "Dodaj"
        buttonAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = editName.getText().toString().trim();
                String city = editCity.getText().toString().trim();
                if (!name.isEmpty() && !city.isEmpty()) {
                    boolean inserted = db.addUser(name, city);
                    Toast.makeText(MainActivity.this, inserted ? "Dodano!" : "Błąd!",
Toast.LENGTH_SHORT).show();
                    editName.setText("");
                    editCity.setText("");
                } else {
                    Toast.makeText(MainActivity.this, "Uzupełnij wszystkie pola!",
Toast.LENGTH_SHORT).show();
                }
            }
        });

        // 🔍 Przycisk "Szukaj"
        buttonSearch.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String query = editSearch.getText().toString().trim();
                String column = spinnerSearchBy.getSelectedItem().toString().equals("Imię") ? "name"
: "city";

                Cursor cursor = db.searchUsers(column, query);
                StringBuilder builder = new StringBuilder();
                if (cursor.getCount() == 0) {
                    builder.append("Brak wyników.");
                } else {
                    while (cursor.moveToNext()) {
                        String name = cursor.getString(1);
                        String city = cursor.getString(2);
                        builder.append(name).append(" - ").append(city).append("\n");
                    }
                }
                textResults.setText(builder.toString());
                cursor.close();
            }
        });
    }
}
```

# Rejestracja w FireBase

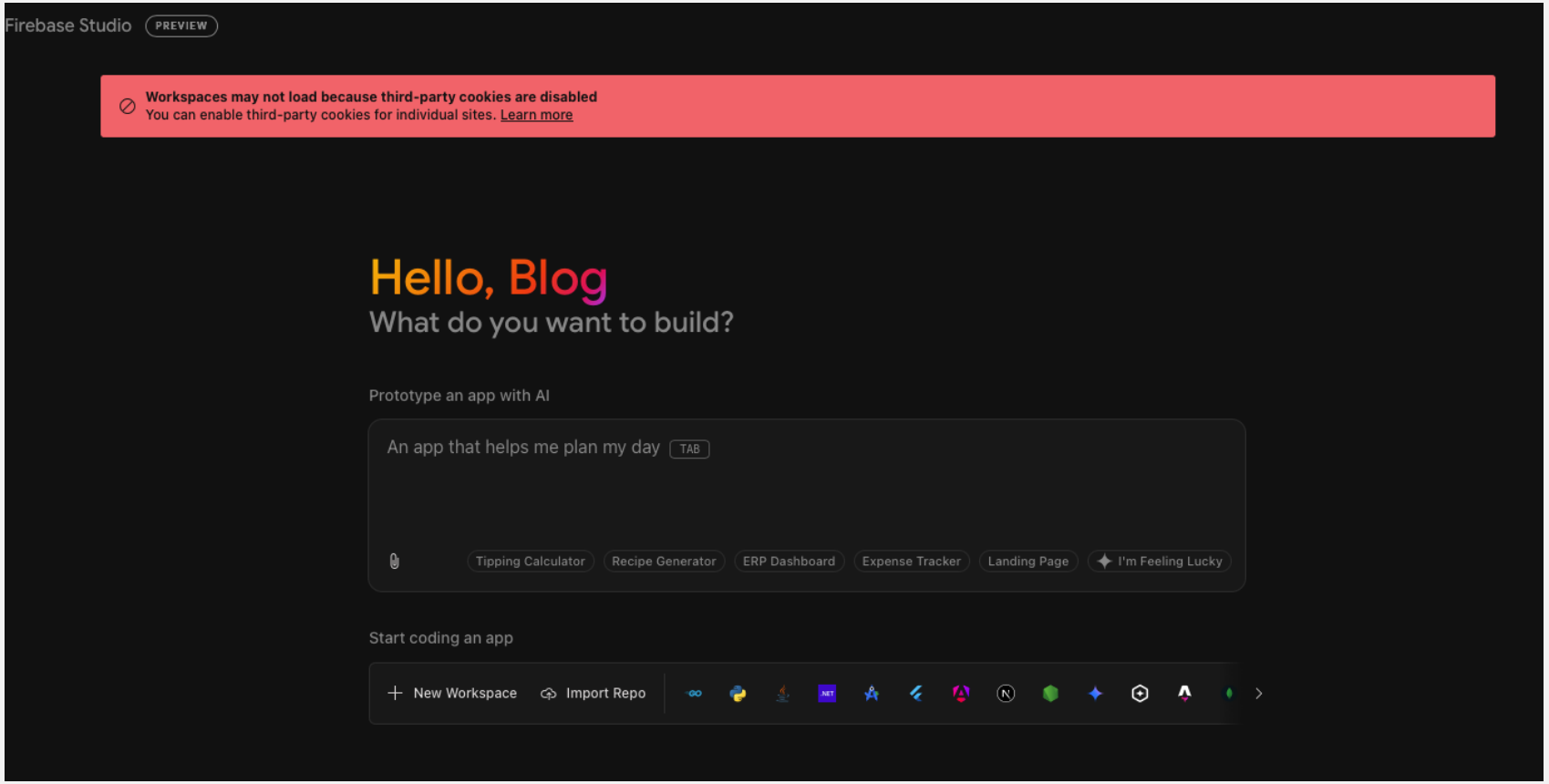


# Co to za wynalazek?

Firebase to platforma stworzona przez Google, która oferuje zestaw narzędzi i usług wspierających tworzenie aplikacji mobilnych i webowych. Ułatwia zarządzanie backendem bez konieczności tworzenia własnych serwerów. Oto główne funkcje Firebase:

1. **Firebase Authentication**
    1. Obsługuje logowanie użytkowników przez e-mail, hasło, numer telefonu, konta Google, Facebook, Twitter itd.
    2. Ułatwia rejestrację, logowanie, resetowanie hasła oraz zarządzanie sesjami.
2. **Firebase Realtime Database**
    1. Baza danych NoSQL, która przechowuje dane w czasie rzeczywistym.
    2. Zmiany danych są automatycznie synchronizowane z klientami.
3. **Cloud Firestore**
    1. Nowsza i bardziej rozbudowana baza danych NoSQL niż Realtime Database.
    2. Obsługuje zapytania, filtrowanie, sortowanie i synchronizację offline.
4. **Firebase Storage**
    1. Umożliwia przechowywanie i pobieranie plików (np. obrazków, filmów).
    2. Przydatne np. do dodawania zdjęć profilowych lub załączników.
5. **Firebase Cloud Messaging (FCM)**
    1. Narzędzie do wysyłania powiadomień push do aplikacji.
6. **Firebase Hosting**
    1. Hosting dla aplikacji webowych i stron internetowych.
7. **Firebase Crashlytics**
    1. Monitorowanie błędów i awarii aplikacji w czasie rzeczywistym.
8. **Firebase Analytics**
    1. Darmowa analiza użytkowania aplikacji (np. liczba użytkowników, czas korzystania, lokalizacja).
9. **Firebase Functions**
    1. Pozwala tworzyć funkcje backendowe w chmurze, które uruchamiają się w odpowiedzi na zdarzenia (np. zapis danych, rejestracja użytkownika).

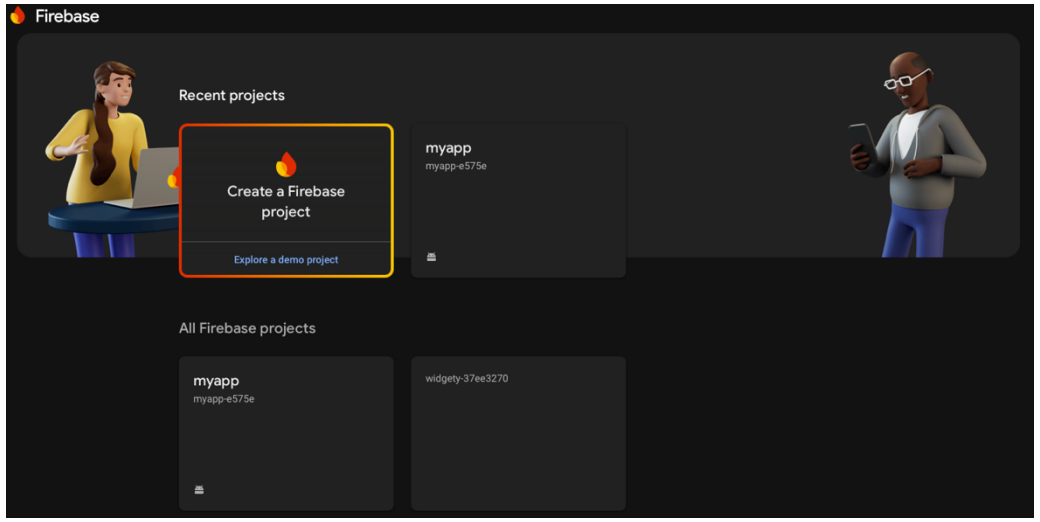
Firebase świetnie integruje się z Androidem, iOS i webem, co czyni go popularnym wyborem dla programistów, którzy chcą szybko stworzyć funkcjonalną aplikację bez budowania całego zaplecza od podstaw.

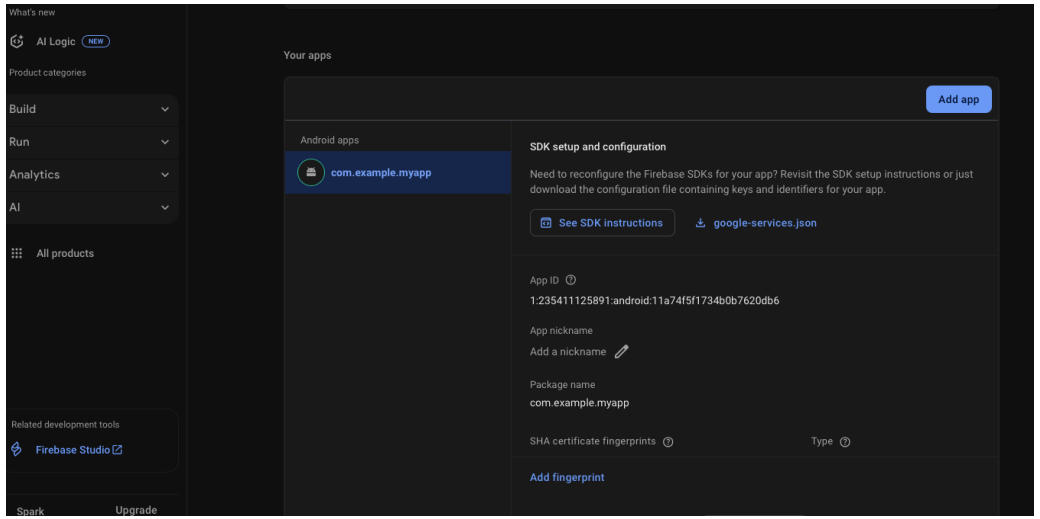



# Firebase to również AI

# Jak to zrobić?

Po prosty wybieramy z listy jakiego rodzaju project potrzebujemy, następnie generujemy plik JSON zawierający potrzebne narzędzia i wklejamy do naszego projektu. W ten sposób możemy zrobić np. zewnętrzną bazę danych i podłączyć do niej wielu użytkowników.



# RecycledView – do wyświetlenia dużych kolekcji danych

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

**Holder i Adapter w jednym pliku.**

```
public class CarAdapter extends RecyclerView.Adapter<CarAdapter.CarViewHolder> {
    private List<Car> carList;

    public static class CarViewHolder extends RecyclerView.ViewHolder {
        TextView carTitle, carDetails;

        public CarViewHolder(View itemView) {
            super(itemView);
            carTitle = itemView.findViewById(R.id.carTitle);
            carDetails = itemView.findViewById(R.id.carDetails);
        }
    }

    @Override
    public CarViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_car, parent, false);
        return new CarViewHolder(view);
    }

    @Override
    public void onBindViewHolder(CarViewHolder holder, int position) {
        Car car = carList.get(position);
        holder.carTitle.setText(car.getBrand() + " " + car.getModel());
        holder.carDetails.setText("Rok: " + car.getYear() + ", Cena: " + car.getPrice());
    }

    @Override
    public int getItemCount() {
        return carList.size();
    }

    public CarAdapter(List<Car> carList) {
        this.carList = carList;
    }
}
```

# RecycledView – do wyświetlenia dużych kolekcji danych

**Dla przypomnienia:**

- Adapter zarządza danymi i tworzy widoki.
- ViewHolder przechowuje odniesienia do elementów jednego widoku listy.
- RecyclerView wyświetla listę z użyciem Adaptera i ViewHolderów, recyklingując stare widoki dla lepszej wydajności.

# RecycledView – do wyświetlenia dużych kolekcji danych

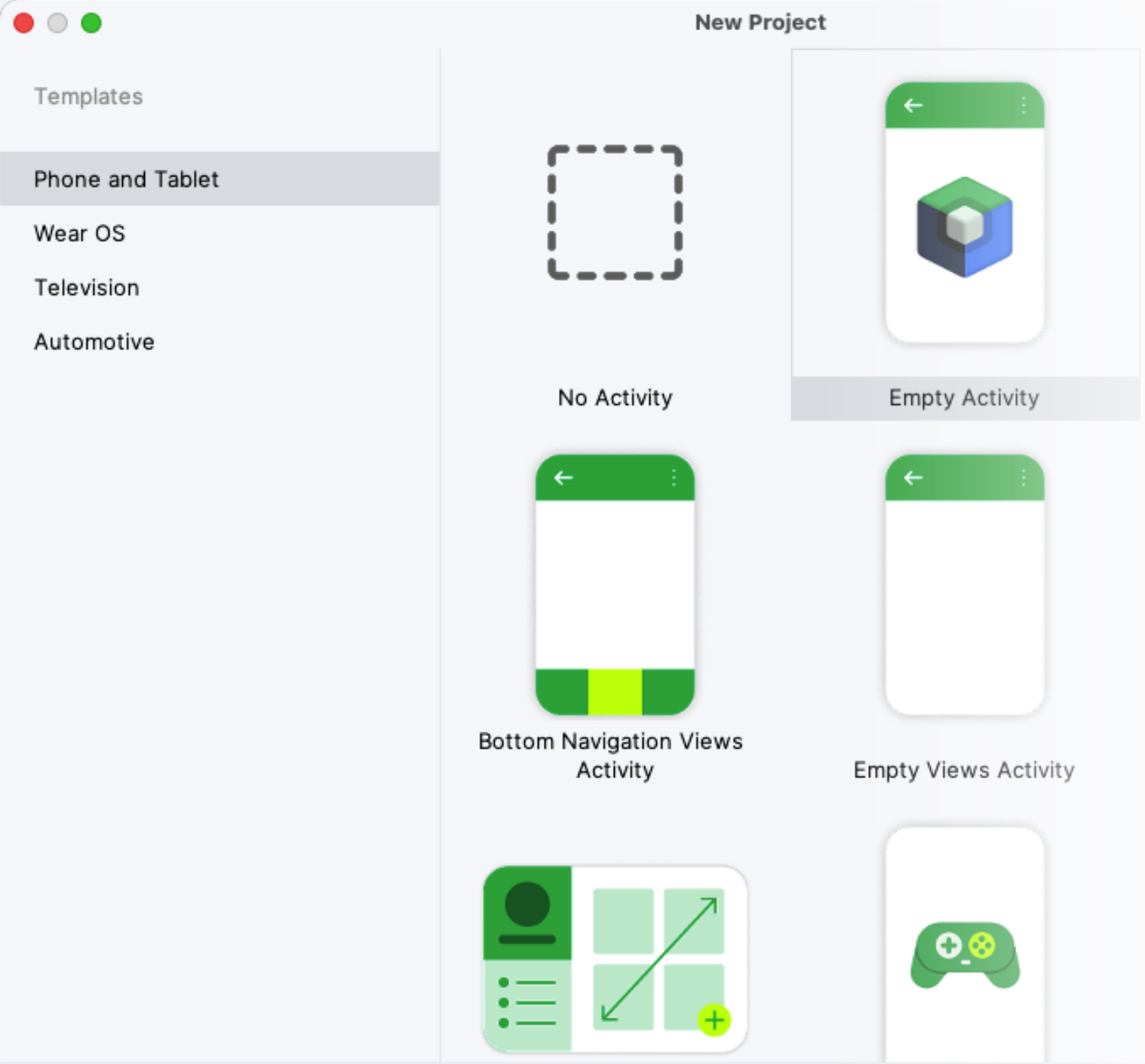
ViewHolder to klasa wewnętrzna adaptera.

Jej zadaniem jest **trzymanie referencji do elementów widoku** (np. TextView, ImageView) dla pojedynczego wiersza listy. Dzięki temu recykling widoków jest szybszy.

… a po ludzku? Jeżeli mamy listę, w której będą wyświetlać się różne dane po aktualizacji to cały czas będziemy mieć taką samą czcionkę, ramki, linie itd..

# RecycledView – do wyświetlenia dużych kolekcji danych

| Cecha | ListView | RecyclerView |
| --- | --- | --- |
| Recykling widoków | Tak, automatyczny | Tak, kontrolowany |
| Elastyczność układu | Ograniczona | Bardzo duża |
| Animacje i gesty | Trudne | Wbudowane |
| Obsługa wielu typów widoków | Uciążliwa | Bardzo łatwa |

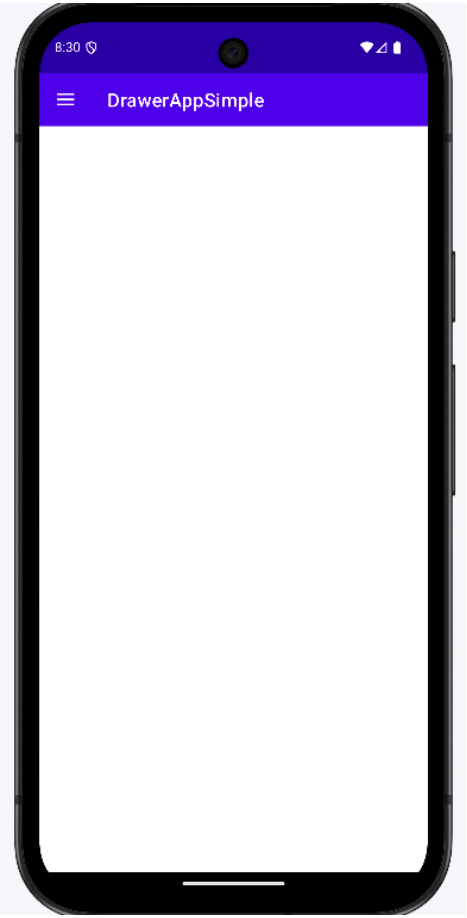


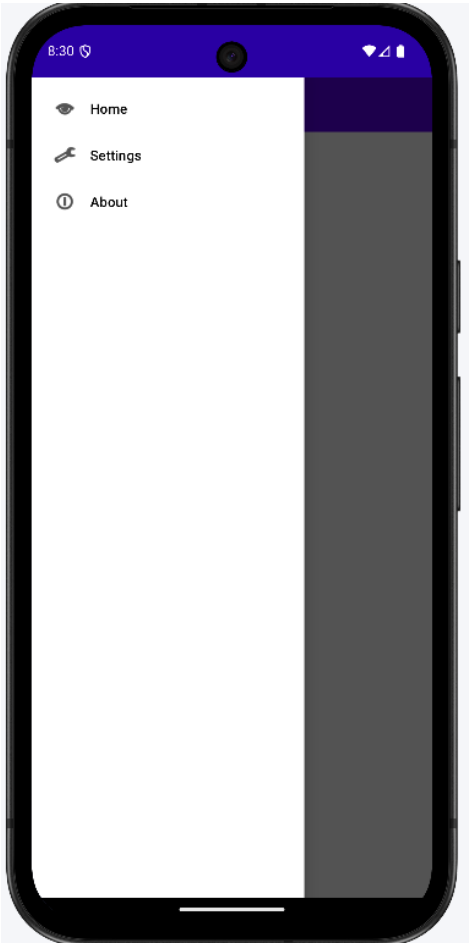

# Dodatkowe Activity do nawigacji

Dotychczasowe aplikacje zawierały podstawowe elementy. Korzystaliśmy z klas Activity, które mogły domyślnie korzystać z języka Java. Kasowaliśmy dane domyślne i wstawialiśmy swój kod.

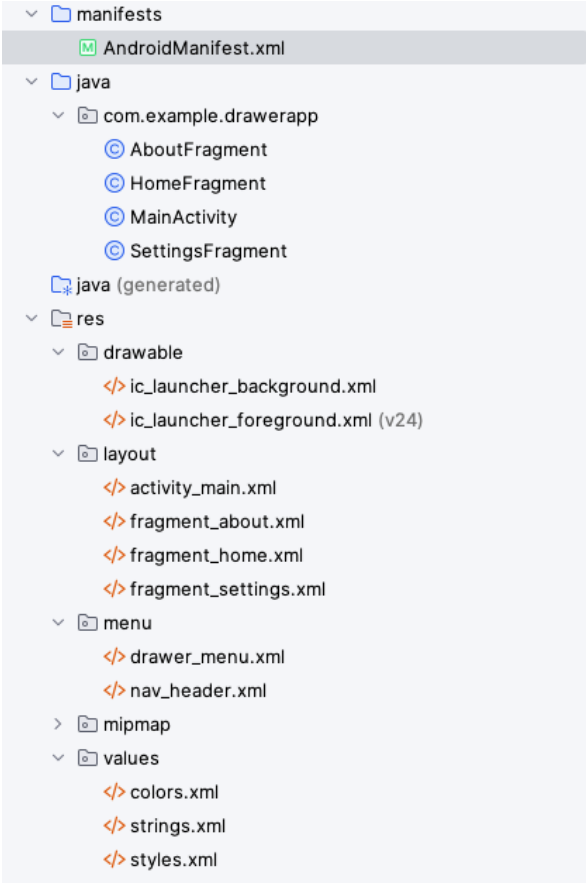
Bardziej zaawansowane aplikacje wykorzystują te klasy jako domyślne szablony, które można rozwijać.

# NavigationDrawActivity – prosty pasek z menu



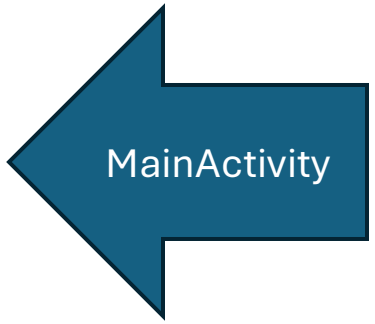




**Prosty wysuwany pasek z przybornikiem może służyć również jako menu.**



# NavigationDrawActivity – prosty pasek z menu

```
package com.example.drawerapp;

import android.os.Bundle;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.GravityCompat;
import androidx.drawerlayout.widget.DrawerLayout;

import com.google.android.material.navigation.NavigationView;

public class MainActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener {

    private DrawerLayout drawerLayout;
    private NavigationView navigationView;
    private Toolbar toolbar;

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

        drawerLayout = findViewById(R.id.drawer_layout);
        navigationView = findViewById(R.id.nav_view);
        toolbar = findViewById(R.id.toolbar);

        setSupportActionBar(toolbar);

        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawerLayout, toolbar,
                R.string.navigation_drawer_open,
                R.string.navigation_drawer_close);

        drawerLayout.addDrawerListener(toggle);
        toggle.syncState();

        navigationView.setNavigationItemSelectedListener(this);
    }

    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem item) {
        int id = item.getItemId();

        if (id == R.id.nav_home) {
            Toast.makeText(this, "Home clicked", Toast.LENGTH_SHORT).show();
        } else if (id == R.id.nav_settings) {
            Toast.makeText(this, "Settings clicked",
Toast.LENGTH_SHORT).show();
        } else if (id == R.id.nav_about) {
            Toast.makeText(this, "About clicked", Toast.LENGTH_SHORT).show();
        } else {
            return false;
        }

        drawerLayout.closeDrawer(GravityCompat.START);
        return true;
    }


    @Override
    public void onBackPressed() {
        if (drawerLayout.isDrawerOpen(GravityCompat.START)) {
            drawerLayout.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed(); // Zamknij aktywność lub co masz domyślnie
        }
    }
}
```



# NavigationDrawActivity – prosty pasek z menu

```
package com.example.drawerapp;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class AboutFragment extends Fragment {

    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                             @Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_about, container, false);
    }
}
```

Plik Fragments – potrzebujemy trzech (dla każdej pozycji menu w pasku).

Wystarczy przekopiować i zmienić nazwę klasy oraz dopisać inne ID z plików XML.

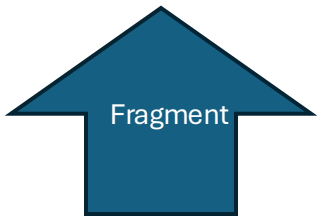
Co to znaczy @nullable? W metodzie możemy zastosować wartość pustą (null) i może być ona zwrócona. Dzięki temu jeżeli będzie konieczne zwrócenie pustej wartości to nie wyskoczy nam żaden błąd.

# NavigationDrawActivity – prosty pasek z menu

## Pliki XML

```
?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res
/android"
  xmlns:tools="http://schemas.android.com/tools"
  xmlns:app="http://schemas.android.com/apk/res-
auto"
  android:id="@+id/drawer_layout"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity">

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <androidx.appcompat.widget.Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"

android:theme="@style/ThemeOverlay.AppCompat.D
ark.ActionBar" />

    <FrameLayout
      android:id="@+id/fragment_container"
      android:layout_width="match_parent"
      android:layout_height="match_parent" />

  </LinearLayout>


<com.google.android.material.navigation.NavigationV
iew
    android:id="@+id/nav_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:fitsSystemWindows="true"
    app:menu="@menu/drawer_menu" />

</androidx.drawerlayout.widget.DrawerLayout>
```

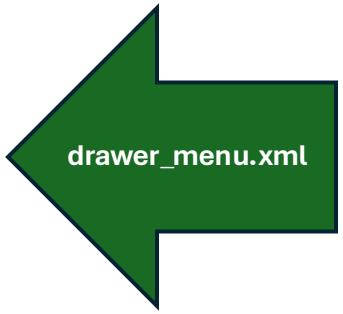
```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/an
droid"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="#FFEFEFEF"
  android:padding="16dp">

  <TextView
    android:id="@+id/about_text"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="To jest ekran O aplikacji"
    android:textSize="18sp" />

</FrameLayout>
```

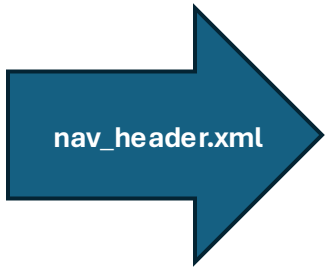


# NavigationDrawActivity – prosty pasek z menu

## Pliki XML

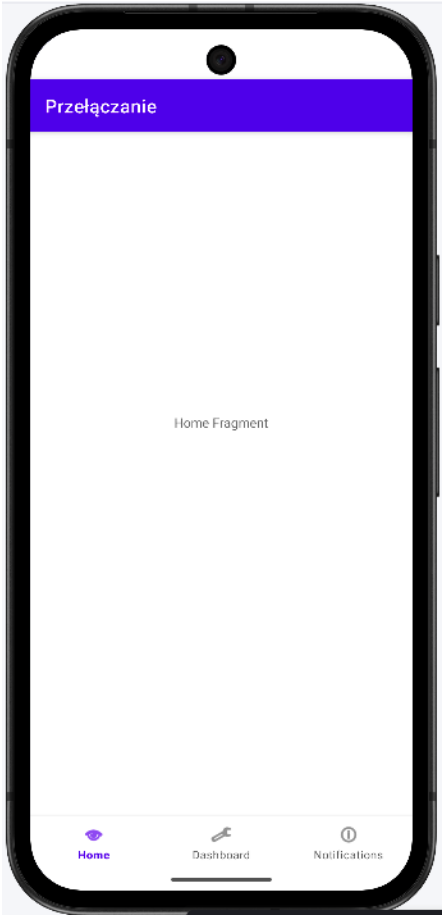
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/nav_home"
    android:icon="@android:drawable/ic_menu_view"
    android:title="Home" />
  <item
    android:id="@+id/nav_settings"
    android:icon="@android:drawable/ic_menu_manage"
    android:title="Settings" />
  <item
    android:id="@+id/nav_about"
    android:icon="@android:drawable/ic_menu_info_details"
    android:title="About" />
</menu>
```

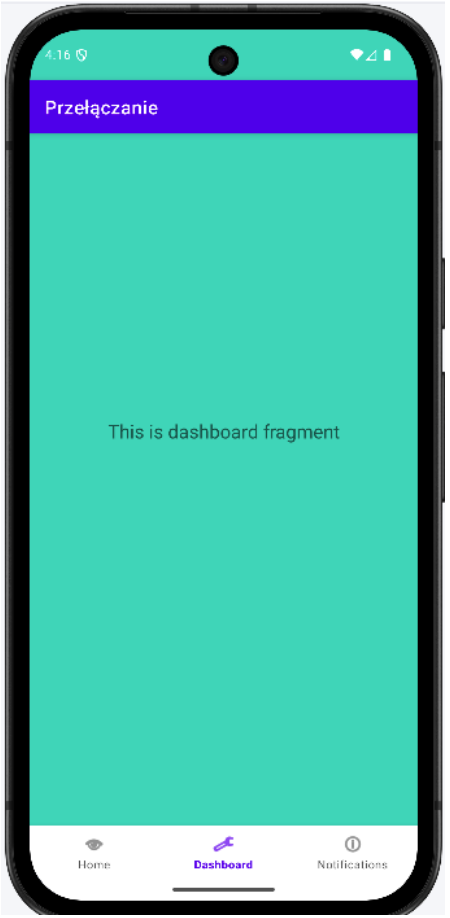


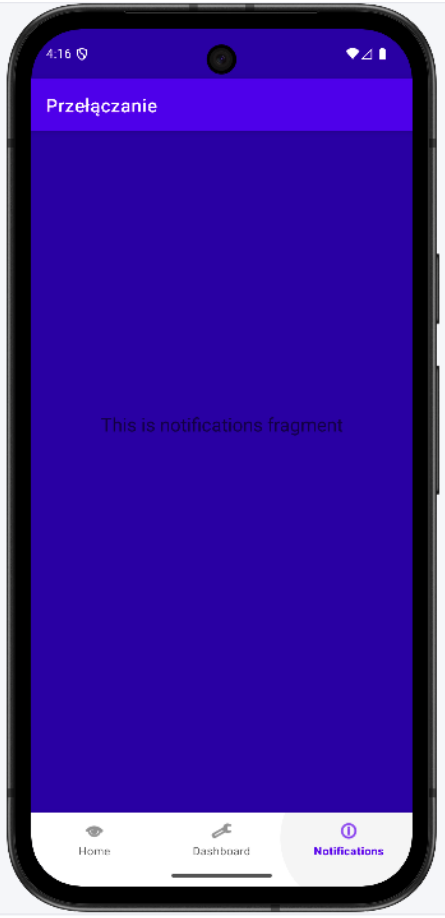



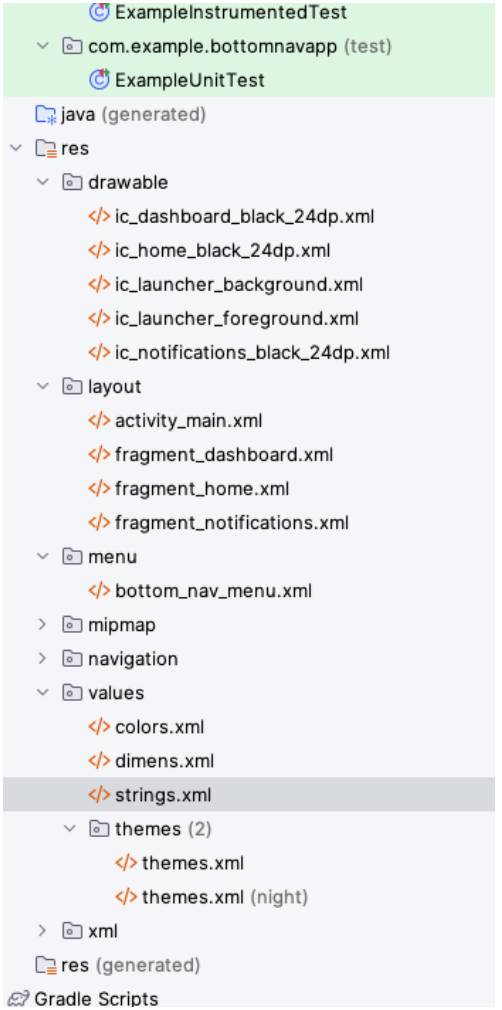

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="160dp"
  android:background="?attr/colorPrimary"
  android:gravity="bottom"
  android:orientation="vertical"
  android:padding="16dp">

  <TextView
    android:id="@+id/nav_header_title"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Witaj w aplikacji!"
    android:textColor="@android:color/white"
    android:textSize="20sp"
    android:textStyle="bold" />

  <TextView
    android:id="@+id/nav_header_subtitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="drawerapp@example.com"
    android:textColor="@android:color/white" />

</LinearLayout>
```

# BottomNavigationActivity – nawigacja na dole aplikacji



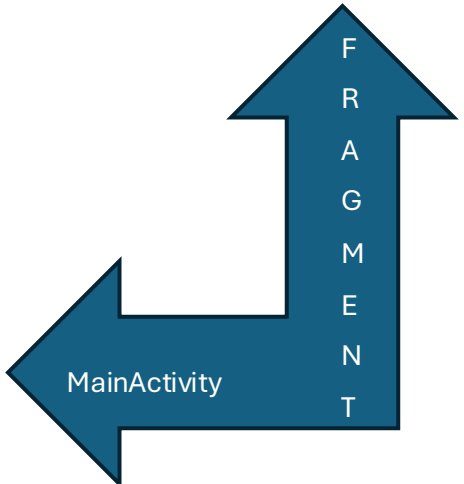



# BottomNavigationActivity – nawigacja na dole aplikacji

```
package com.example.bottomnavapp;

import android.os.Bundle;
import androidx.annotation.NonNull;
import
androidx.appcompat.app.AppCompatActivity;

import
com.example.bottomnavapp.ui.dashboard.D
ashboardFragment;
import
com.example.bottomnavapp.ui.notifications.
NotificationsFragment;
import
com.google.android.material.bottomnavigatio
n.BottomNavigationView;
import androidx.fragment.app.Fragment;

import android.view.MenuItem;

public class MainActivity extends
AppCompatActivity {

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

        BottomNavigationView bottomNav =
findViewById(R.id.bottom_navigation);

bottomNav.setOnItemSelectedListener(navLi
stener);

        // Domyślny fragment
        if (savedInstanceState == null) {

getSupportFragmentManager().beginTransacti
on()
                .replace(R.id.fragment_container,
new HomeFragment())
                .commit();
        }
    }

    private final
BottomNavigationView.OnItemSelectedListen
er navListener =
            new
BottomNavigationView.OnItemSelectedListen
er() {
                @Override
                public boolean
onNavigationItemSelected(@NonNull
MenuItem item) {
                    Fragment selectedFragment = null;

                    int id = item.getItemId();
                    if (id == R.id.nav_home) {
                        selectedFragment = new
HomeFragment();
                    } else if (id == R.id.nav_dashboard) {
                        selectedFragment = new
DashboardFragment();
                    } else if (id == R.id.nav_notifications) {
                        selectedFragment = new
NotificationsFragment();
                    }


getSupportFragmentManager().beginTransacti
on()

.replace(R.id.fragment_container,
selectedFragment)
                            .commit();
                    return true;
                }
            };
}
```

```
package com.example.bottomnavapp;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import androidx.fragment.app.Fragment;

public class HomeFragment extends Fragment {
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        return inflater.inflate(R.layout.fragment_home, container, false);
    }
}
```



# BottomNavigationActivity – nawigacja na dole aplikacji

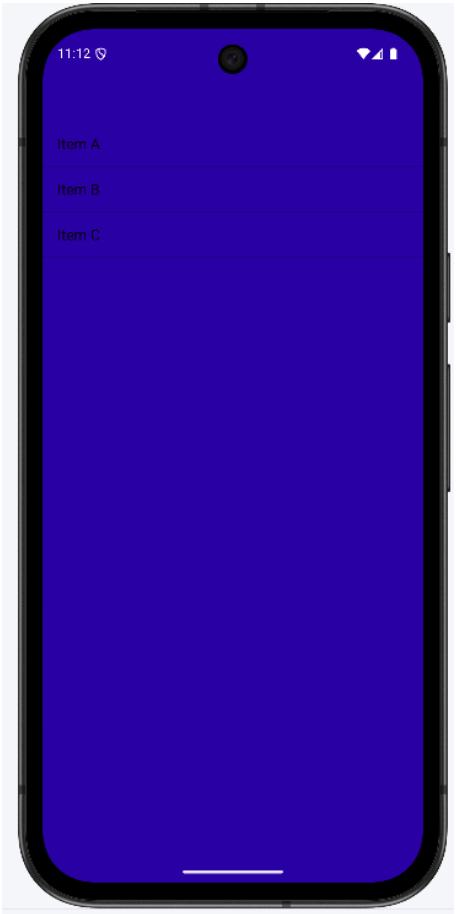
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <androidx.fragment.app.FragmentContainerView
      android:id="@+id/fragment_container"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:name="com.example.bottomnavapp.HomeFragment"
/>


<com.google.android.material.bottomnavigation.BottomNavigation
View
      android:id="@+id/bottom_navigation"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      app:menu="@menu/bottom_nav_menu"
      app:labelVisibilityMode="labeled"
      android:background="?android:attr/windowBackground" />
</LinearLayout>
```

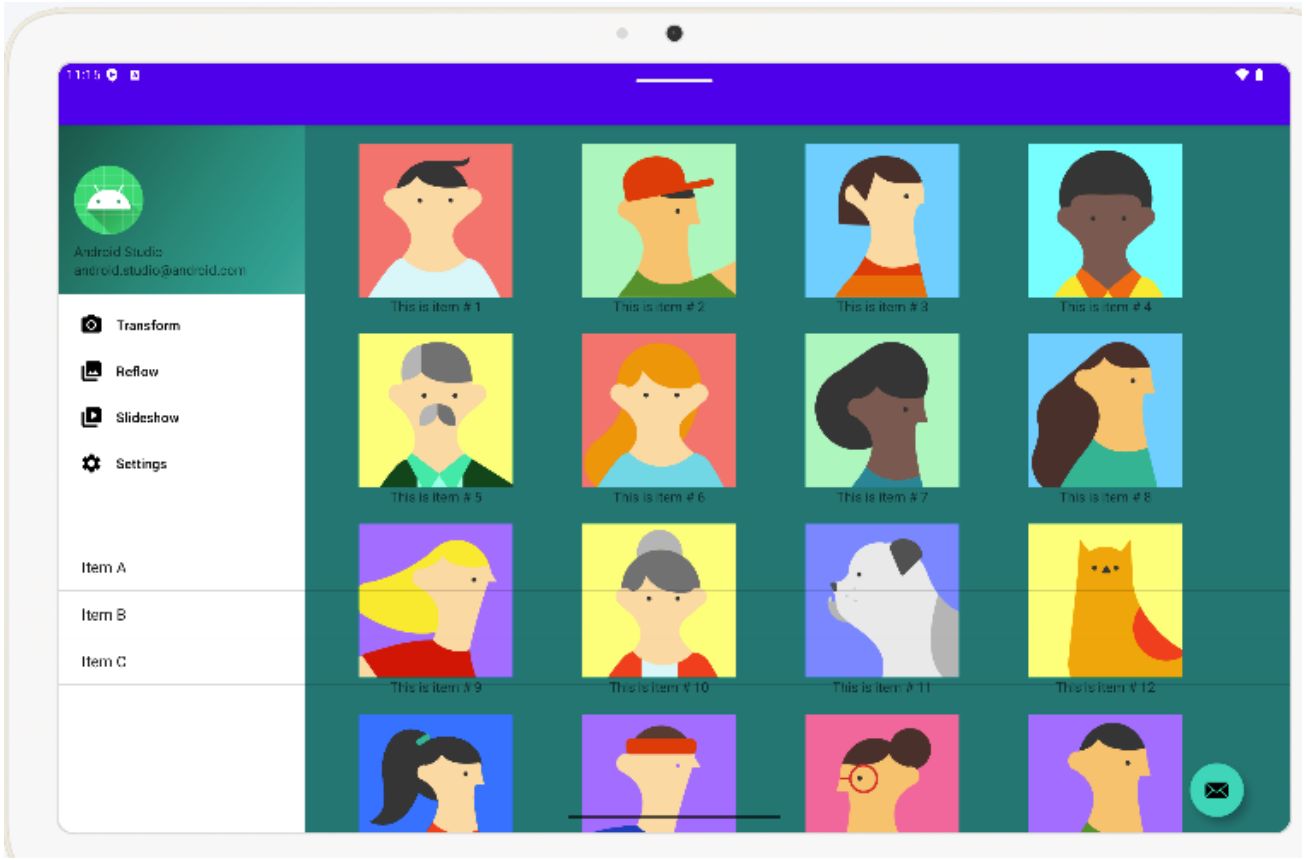
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">


   <androidx.fragment.app.FragmentContainerView
      android:id="@+id/fragment_container"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:name="com.example.bottomnavapp.HomeFragment" />


   <com.google.android.material.bottomnavigation.BottomNavigationView
      android:id="@+id/bottom_navigation"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      app:menu="@menu/bottom_nav_menu"
      app:labelVisibilityMode="labeled"
      android:background="?android:attr/windowBackground" />
</LinearLayout>
```

# BottomNavigationActivity – nawigacja na dole aplikacji (Plik XML z Bottom Navigation)

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android"
>
  <item
    android:id="@+id/nav_home"
    android:icon="@android:drawable/ic_menu_view"
    android:title="Home" />
  <item
    android:id="@+id/nav_dashboard"
    android:icon="@android:drawable/ic_menu_manage"
    android:title="Dashboard" />
  <item
    android:id="@+id/nav_notifications"
    android:icon="@android:drawable/ic_menu_info_details"
    android:title="Notifications" />
</menu>
```

# Responsive ViewActivity – dostosowanie wyglądu dla różnych urządzeń

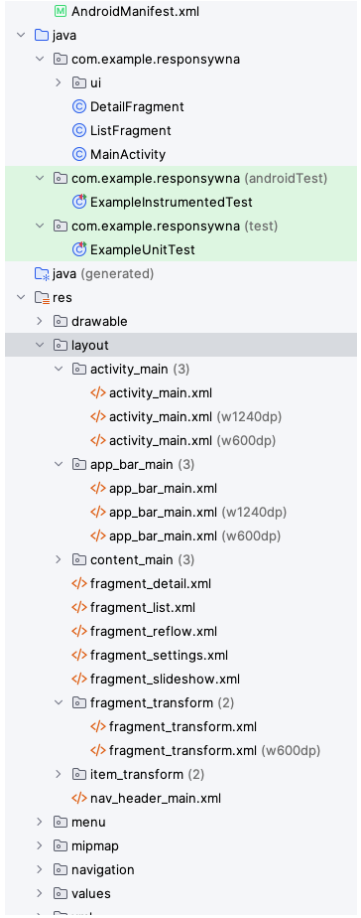



?

Tak, to jest ta sama aplikacja!

# Responsive ViewActivity – dostosowanie wyglądu dla różnych urządzeń




```
package com.example.responsywna;

import android.os.Bundle;
import android.view.MenuItem;
import android.view.Menu;

import
com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.navigation.NavigationView;

import androidx.annotation.NonNull;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.fragment.NavHostFragment;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.appcompat.app.AppCompatActivity;

import com.example.responsywna.databinding.ActivityMainBinding;

public class MainActivity extends AppCompatActivity implements
ListFragment.OnItemSelectedListener {
  private boolean isTablet;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Ładowanie odpowiedniego layoutu
    setContentView(R.layout.activity_main);

    isTablet = findViewById(R.id.detail_container) != null;

    if (savedInstanceState == null) {
      ListFragment listFragment = new ListFragment();
      if (isTablet) {
        getSupportFragmentManager().beginTransaction()
            .replace(R.id.list_container, listFragment)
            .commit();
      } else {
        getSupportFragmentManager().beginTransaction()
            .replace(R.id.container, listFragment)
            .commit();
      }
    }
  }

  @Override
  public void onItemSelected(String item) {
    DetailFragment detailFragment = DetailFragment.newInstance(item);
    if (isTablet) {
      getSupportFragmentManager().beginTransaction()
          .replace(R.id.detail_container, detailFragment)
          .commit();
    } else {
      getSupportFragmentManager().beginTransaction()
          .replace(R.id.container, detailFragment)
          .addToBackStack(null)
          .commit();
    }
  }
}
```

# Responsive ViewActivity – dostosowanie wyglądu dla różnych urządzeń

## Fragmenty:

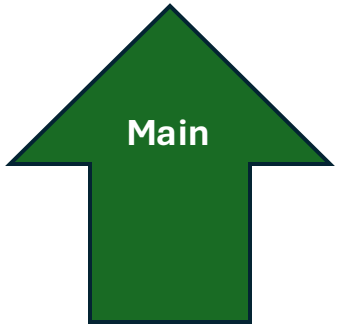
```
package com.example.responsywna;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.fragment.app.Fragment;

import org.jetbrains.annotations.Nullable;

public class DetailFragment extends Fragment {

    private static final String ARG_ITEM = "item";

    public static DetailFragment newInstance(String item) {
        DetailFragment fragment = new DetailFragment();
        Bundle args = new Bundle();
        args.putString(ARG_ITEM, item);
        fragment.setArguments(args);
        return fragment;
    }

    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_detail, container, false);
        TextView textView = view.findViewById(R.id.detail_text);

        if (getArguments() != null) {
            String item = getArguments().getString(ARG_ITEM);
            textView.setText("Szczegóły: " + item);
        }

        return view;
    }
}
```

```
package com.example.responsywna;

import android.content.Context;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ListView;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class ListFragment extends Fragment {

    public interface OnItemSelectedListener {
        void onItemSelected(String item);
    }

    private OnItemSelectedListener listener;

    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        listener = (OnItemSelectedListener) context;
    }

    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_list, container, false);
        ListView listView = view.findViewById(R.id.list);

        String[] items = {"Item A", "Item B", "Item C"};

        listView.setAdapter(new ArrayAdapter<>(requireContext(),
                android.R.layout.simple_list_item_1, items));

        listView.setOnItemClickListener((parent, view1, position, id) ->

                listener.onItemSelected(items[position])
        );

        return view;
    }
}
```

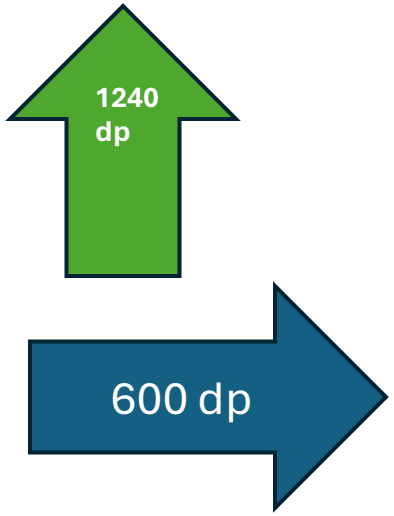
# Responsive ViewActivity – dostosowanie wyglądu dla różnych urządzeń

## XML z Activity Main

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res
/android"
  android:id="@+id/container"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingTop="100dp"
  android:background="@color/purple_700"/>
```



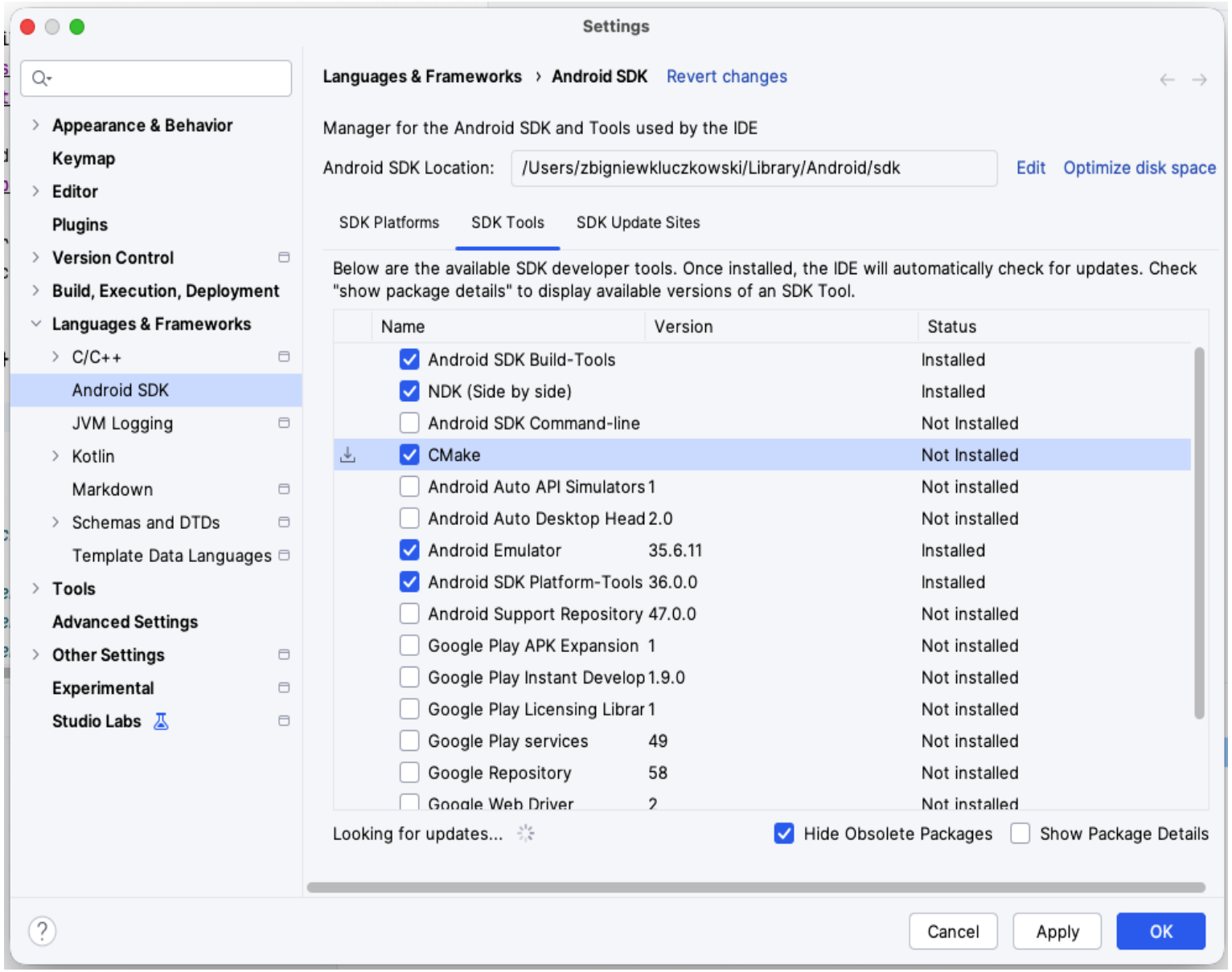

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/container"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@color/teal_700">

  <include
    android:id="@+id/app_bar_main"
    layout="@layout/app_bar_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
</FrameLayout>
```

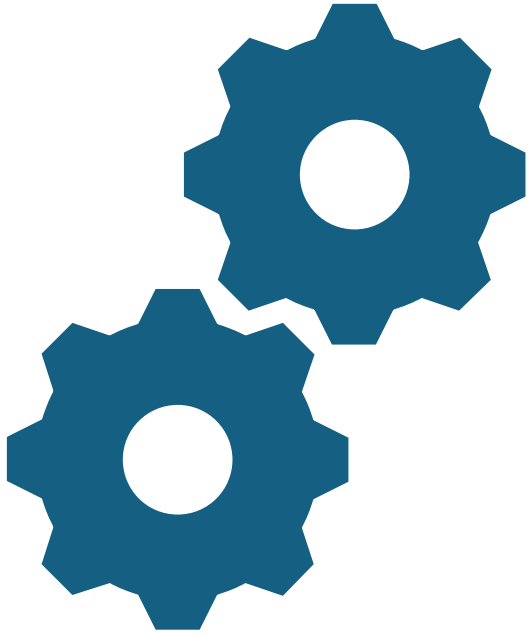



```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="horizontal"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:id="@+id/container"
  android:background="@color/purple_500"
  >

  <FrameLayout
    android:id="@+id/list_container"
    android:layout_width="0dp"
    android:layout_weight="1"
    android:layout_height="match_parent" />

  <FrameLayout
    android:id="@+id/detail_container"
    android:layout_width="0dp"
    android:layout_weight="2"
    android:layout_height="match_parent" />
</LinearLayout>
```

# GameActivity z wykorzystaniem C++

Korzystanie z tej aktywności daje możliwość programowania w C++. Konieczne są zmiany w uprawnieniach i w Gradle.





# GameActivity z wykorzystaniem C++

Dodaj w local.properties:

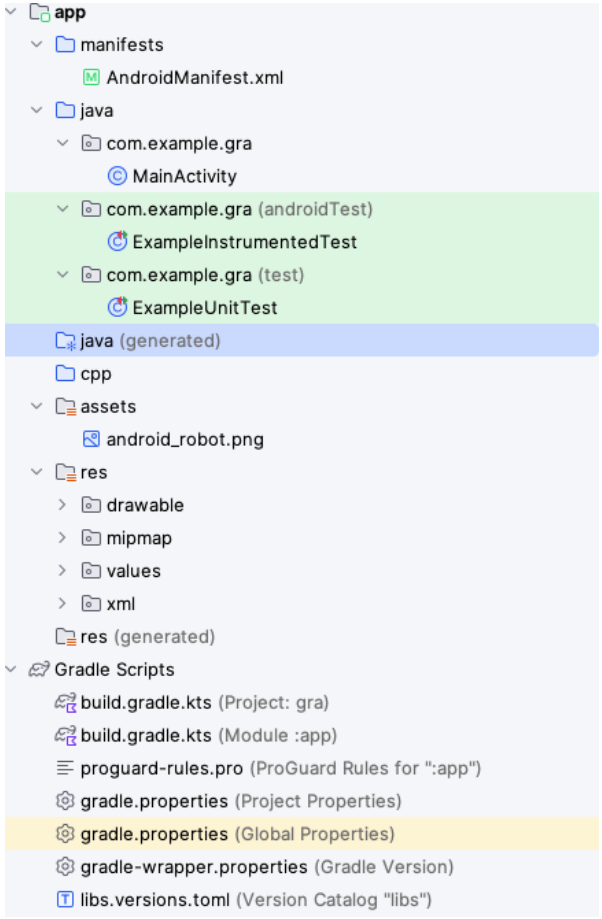
cmake.dir=/Users/zbigniewkluczkowski/Library/Android/sdk/cmake/3.22.1

ndk.dir=/Users/zbigniewkluczkowski/Library/Android/sdk/ndk/25.2.9519653

Co to za „wynalazki”?

Android Studio sam z siebie nie będzie kompilował ani interpretował kodu w C lub w C++. Potrzebuje do tego „tłumacza” w postaci NDK a CMake pozwala pisać w C++ wewnątrz Android Studio. W szerszym rozumieniu oznacza to natywne wspieranie C++.

# GameActivity z wykorzystaniem C++



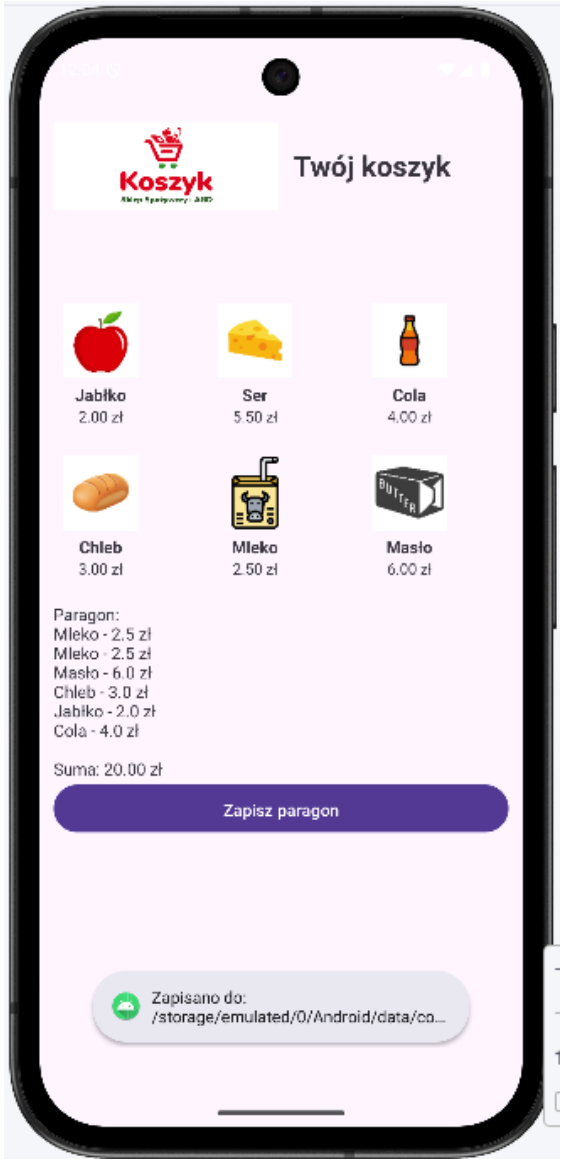

W praktyce ...

Piszesz i umieszczasz pliki w folderze CPP. Następnie są one konwertowane automatycznie do Javy. Pliki XML należy napisać jak w przypadku Javy.

```
<manifest
xmlns:android="http://schemas.android.com/apk
/res/android"
  package="com.example.gra">

  <application
    android:hasCode="false"
    android:label="GameActivity"

android:theme="@android:style/Theme.NoTitleBa
r.Fullscreen">

    <activity
      android:name="android.app.GameActivity"
      android:label="GameActivity"
      android:exported="true"

android:configChanges="orientation|screenSize"
      android:launchMode="singleTask">

      <meta-data
        android:name="android.app.lib_name"
        android:value="native-lib" />

      <intent-filter>
        <action
android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCH
ER" />
      </intent-filter>
    </activity>
  </application>

</manifest>
```

# Podsumowanie - pytania

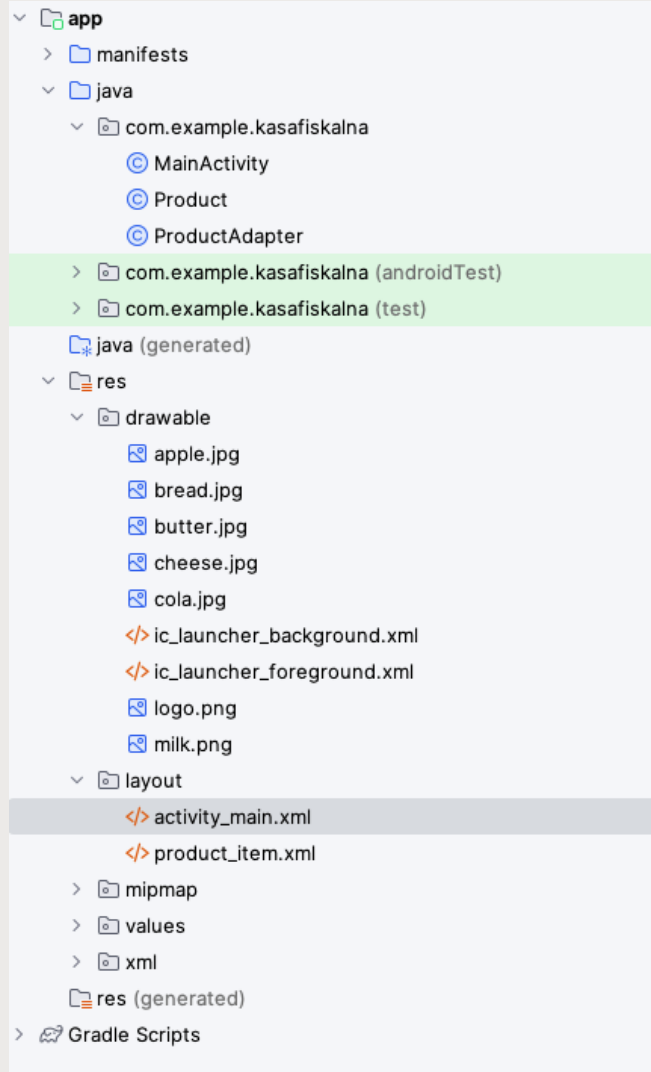
1. **W jakim formacie najczęściej przechowywane są dane w aplikacji?**
2. **Do czego służy Corner:AndroidRadius?**
3. **Do czego służy BufferedReader?**
4. **Jakiego rodzaju bazy danych można umieszczać w projekcie?**
5. **Do czego służy Firebase?**
6. **Czym jest Helper i co zawiera?**
7. **Czy widok w ResponsiveViewActivity może różnić się na wielu urządzeniach?**
8. **Do czego służy CMake i NDK?**




# Zadania do samodzielnego wykonania

## Zadanie 1

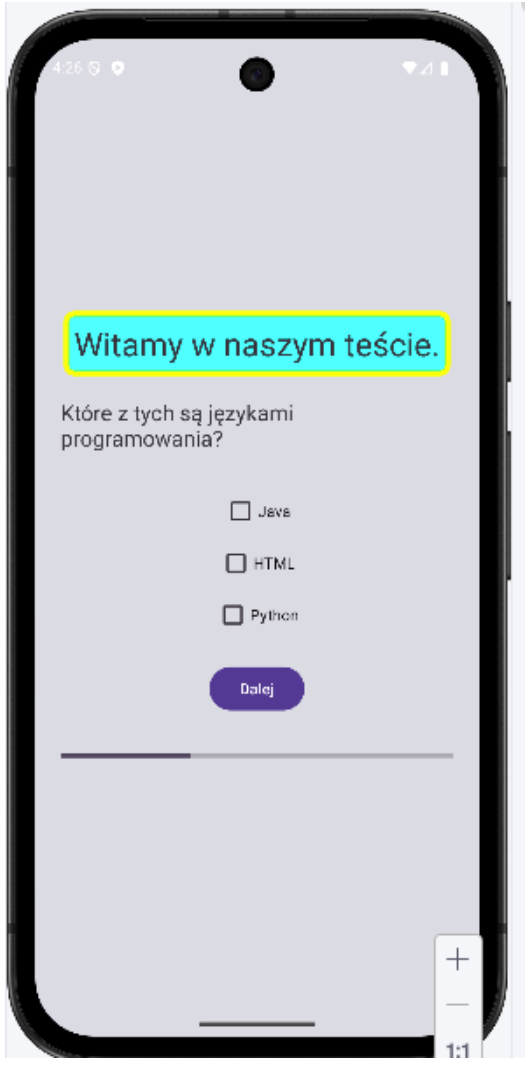
Sklep „koszyk” oferuje szeroki zestaw produktów. Twoim zadaniem jest wykonanie aplikacji kasy fiskalnej z zapisem do pliku według poniższego wzoru.



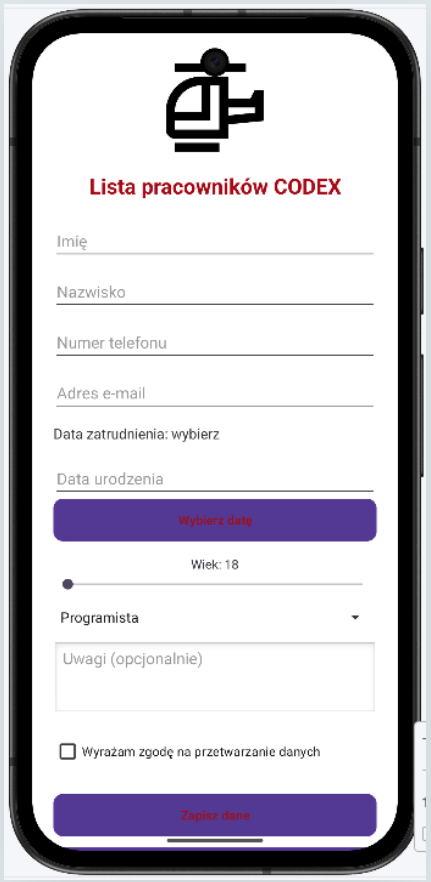
# Zadania do samodzielnego wykonania

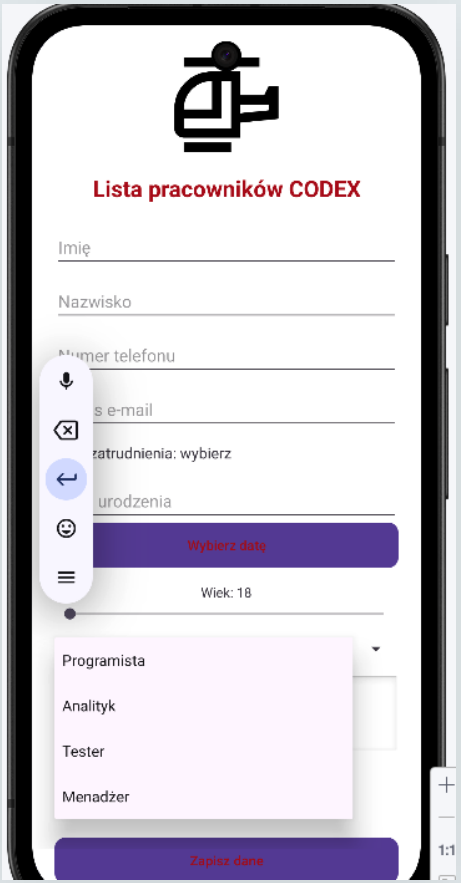
## Zadanie 2

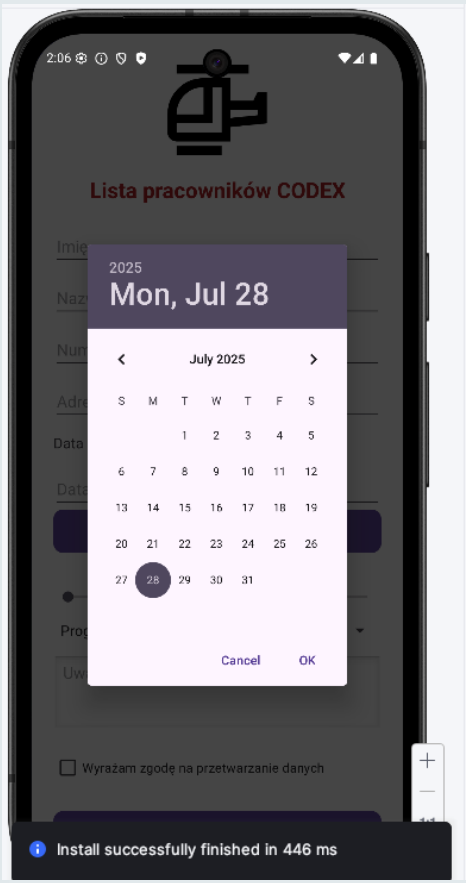
Aplikacja w formie testu. Użytkownik odpowiada na pytania z wieloma poprawnymi odpowiedziami. Po naciśnięciu przycisku odpowiedź zostaje zaakceptowana a następnie przechodzimy do następnego pytania. Aplikacja zawiera ProgressBar oraz zapis wyniku do pliku.




# Zadania do samodzielnego wykonania

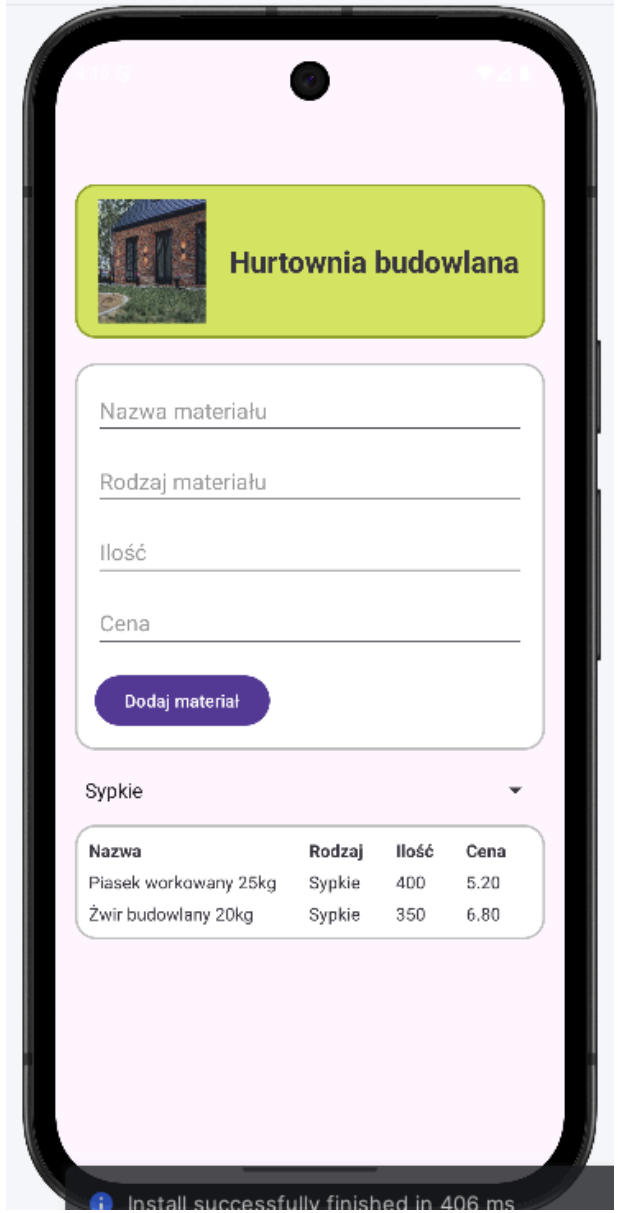



## Zadanie 3

Pracownik wypełnia formularz. Formularz zawiera pole typu data otwierane za pomocą przycisku. Po wybraniu daty wartość zostaje dodana do etykiety. Suwak zawiera wartości od 18 do 80. Użytkownik wybiera jeden z przedstawionych zawodów. Dane zapisują się do pliku.



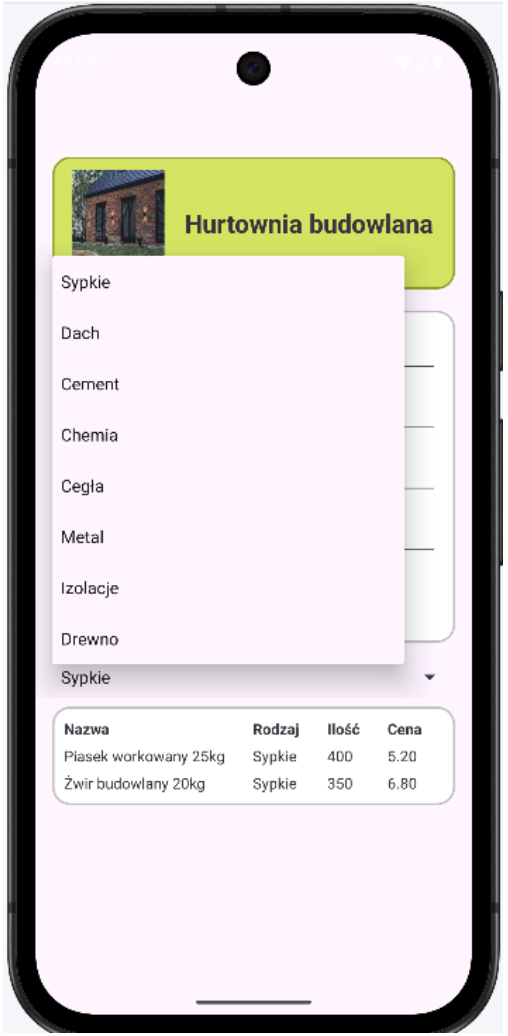

# Zadania do samodzielnego wykonania

## Zadanie 4

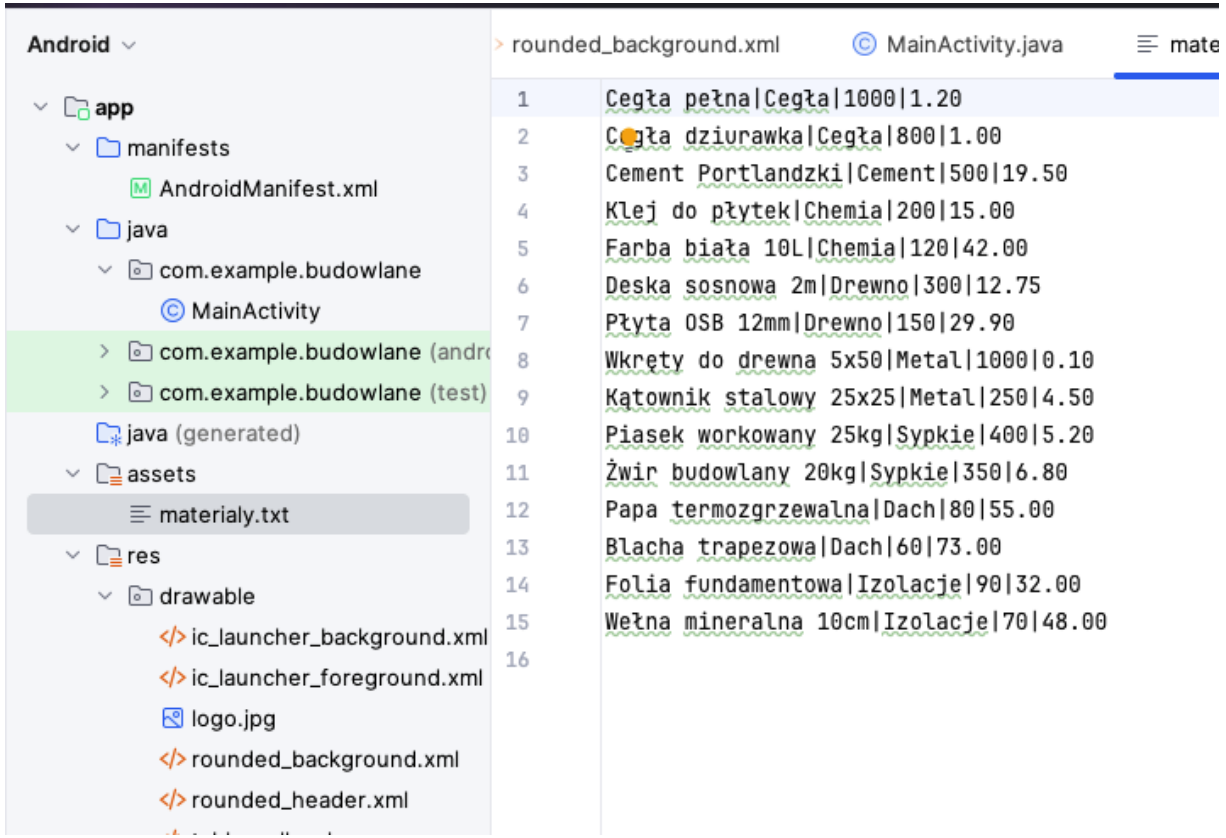
Zadanie w formie aplikacji do zarządzania magazynem budowlanym. Użytkownik może dodawać dane oraz wyświetlać odpowiednie kategorie za pomogą Spinnera. Dane po wyszukaniu wyświetlają się w tabeli

W projekcie zastosowano Corner:AndroidRadius.

# Zadania do samodzielnego wykonania




Zawartość pliku tekstowego

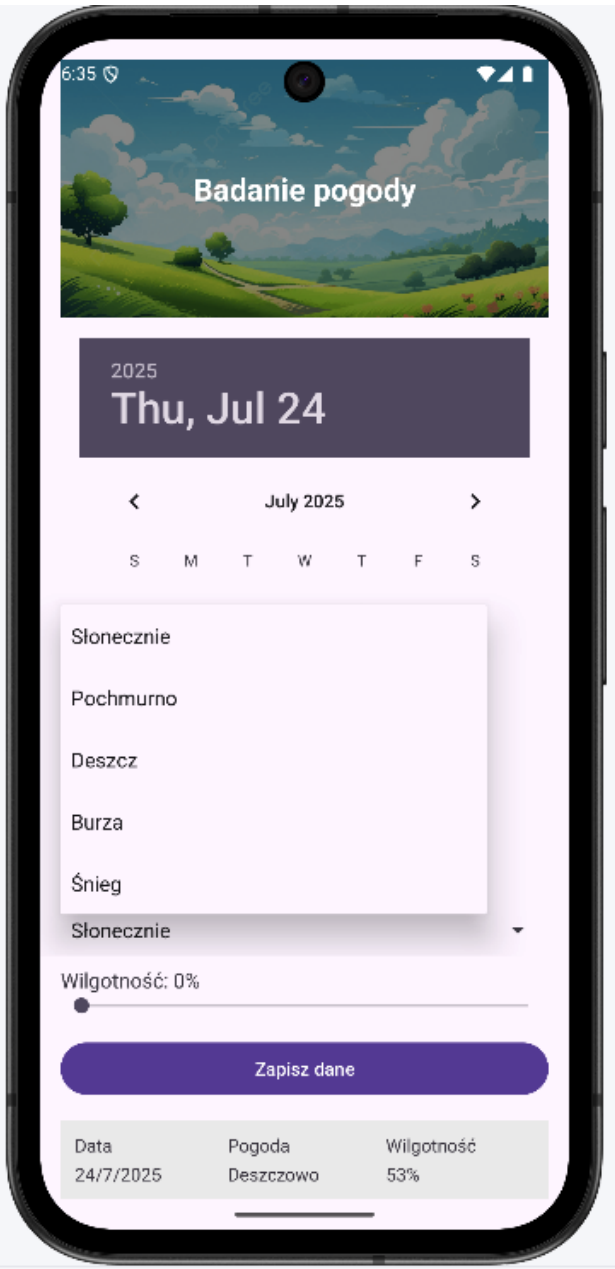


# Zadania do samodzielnego wykonania

## Zadanie 5 (zadanie na 5)

Twoim zadaniem jest wykonanie aplikacji do zbierania danych pogodowych według wzoru zamieszczonego obok.

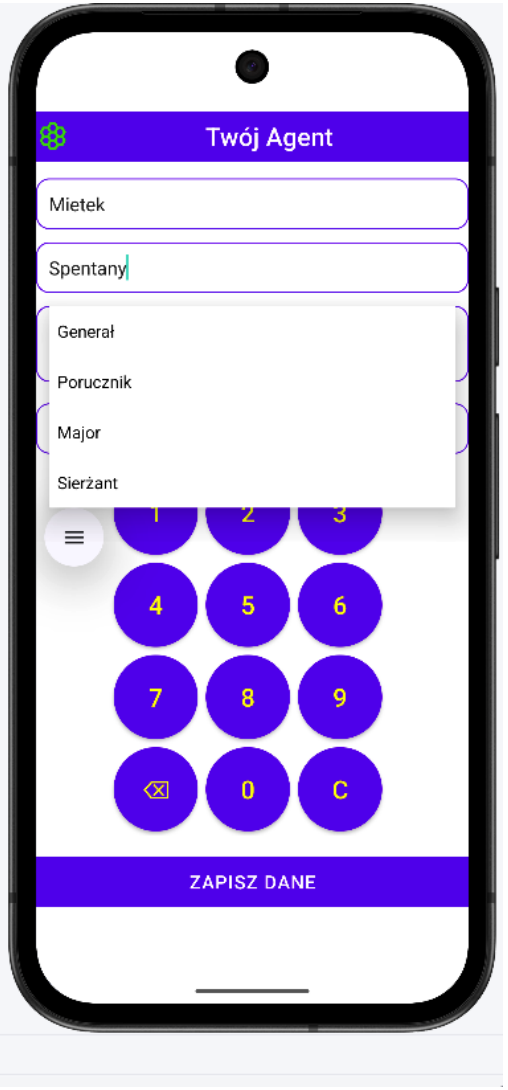
Dane zbierane są w pliku oraz wyświetlane w postaci tabeli.

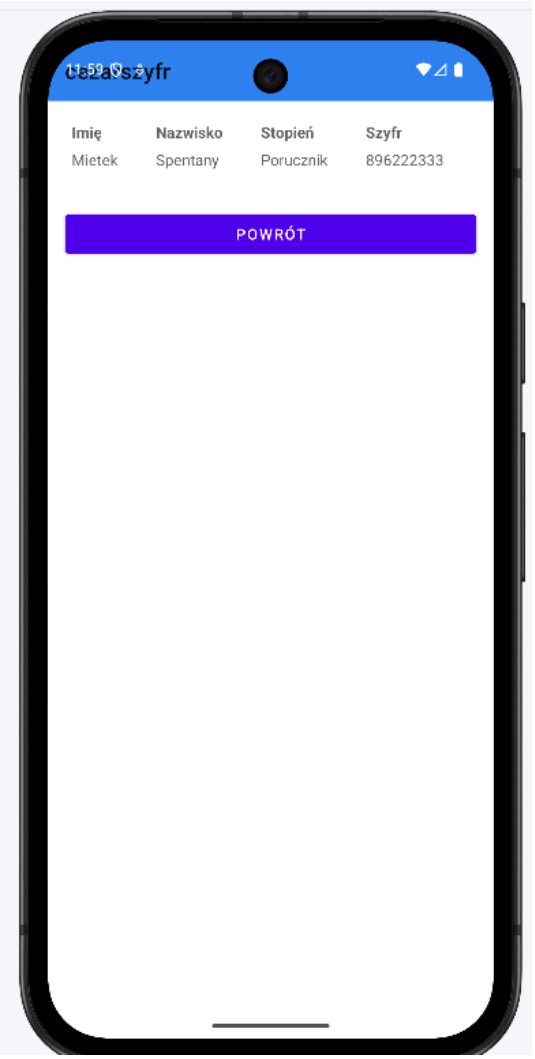



# Zadania do samodzielnego wykonania

## **Zadanie 6** (zadanie na 5)

Aplikacja pobiera od użytkownika dane, w tym numer telefonu wprowadzony klawiatury.  Następnie numer zostaje zaszyfrowany za pomocą szyfru Cezara (+3). Po zapisaniu danych zostają wyświetlone na kolejne Activity w formie tabeli. Druga aktywność zawiera przycisk do powrotu do pierwszej aktywności.

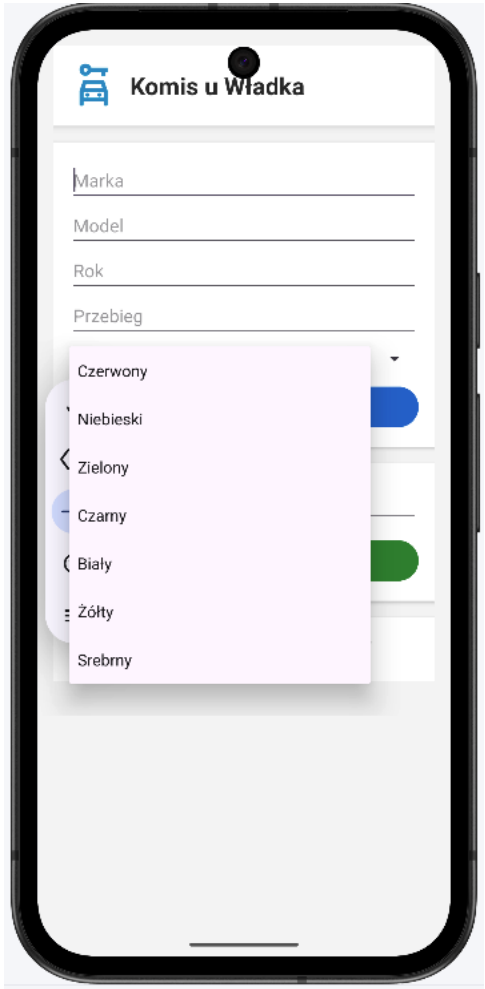


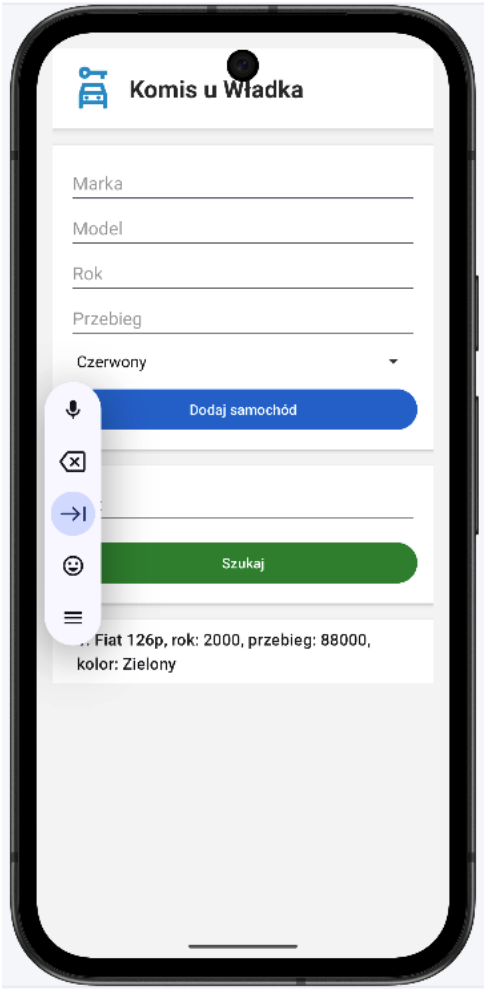



# Zadania do samodzielnego wykonania

## Zadanie 7 (na 5 lub 6)

Użytkownik wpisuje dane do formularza. Dane zapisywane są w bazie danych. Następnie wybiera kolor wśród dostępnych opcji. Wyszukiwarka wyszukuje dane i wyświetla je w postaci listy numerowanej.



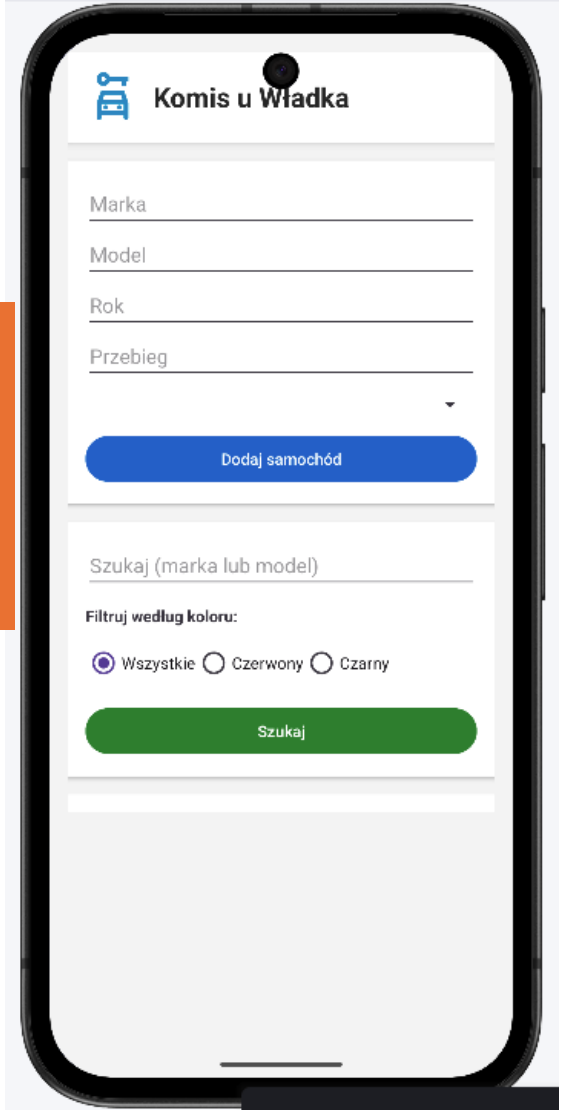



# Zadania do samodzielnego wykonania

```
package com.example.samochody;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBHelper extends SQLiteOpenHelper {

    public static final String DB_NAME = "samochody.db";
    public static final int DB_VERSION = 1;

    public DBHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE auta (" +
                "id INTEGER PRIMARY KEY AUTOINCREMENT, " +
                "marka TEXT, " +
                "model TEXT, " +
                "rok INTEGER, " +
                "przebieg INTEGER, " +
                "kolor TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS auta");
        onCreate(db);
    }

    public boolean dodajAuto(String marka, String model, int rok, int przebieg, String kolor) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues cv = new ContentValues();
        cv.put("marka", marka);
        cv.put("model", model);
        cv.put("rok", rok);
        cv.put("przebieg", przebieg);
        cv.put("kolor", kolor);
        long result = db.insert("auta", null, cv);
        return result != -1;
    }

    public Cursor wyszukajAuto(String query) {
        SQLiteDatabase db = this.getReadableDatabase();
        return db.rawQuery("SELECT * FROM auta WHERE marka LIKE ? OR model LIKE ?", new String[]{"%" + query + "%", "%" + query + "%"});
    }
}
```

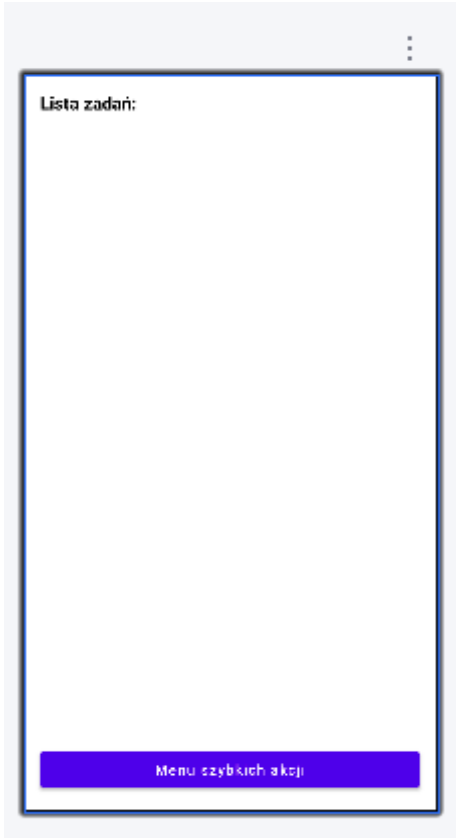
Klasa pomocnicza

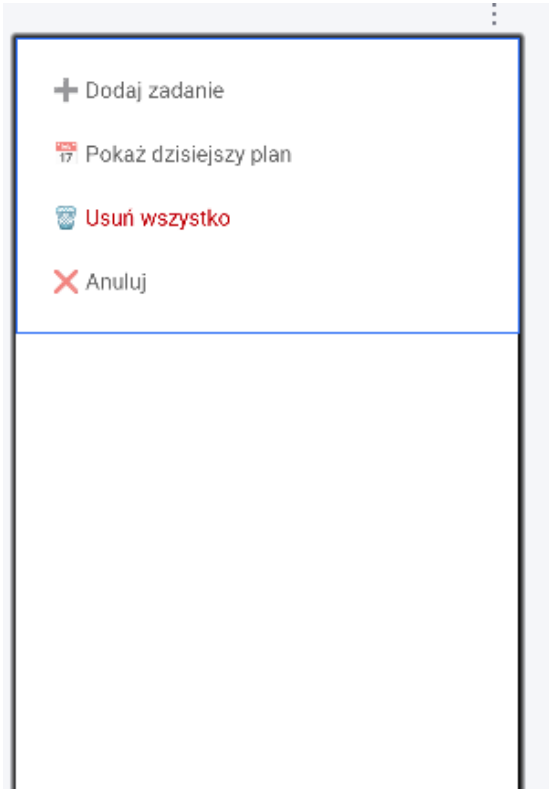
Na najwyższą ocenę rozwiń projekt tak aby można było wyszukiwać samochody po kolorach za pomocą pola typu radio.

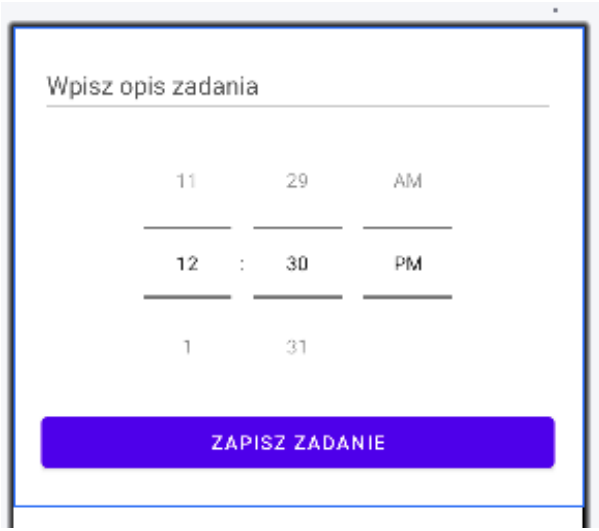


# Zadania do samodzielnego wykonania

## Zadanie 8 (na 5)



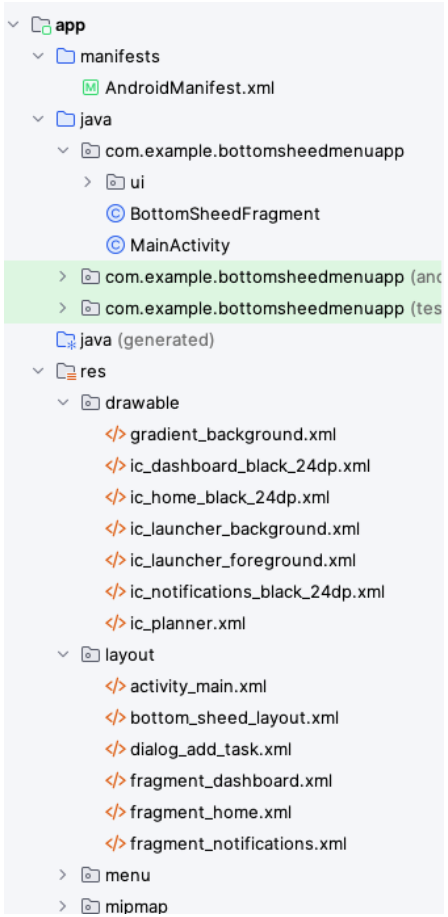




Twoim zadaniem jest wykonanie aplikacji w formie terminarza w zadaniami. Jego działanie opiera się na przycisku oraz BottomNavigationDialog. Po naciśnięciu przycisku wysuwa się menu z opcjami jak na zdjęciu.

Lista zadań wyświetla się na MainActivity. Możesz wybrać pomiędzy listą punktowaną lub numerowaną.

# Zadania do samodzielnego wykonania




W strukturze plików należy umieścić:

- Fragment (znajduje się w nim wysuwane menu z dołu)
- Task (do wprowadzania zadań)
- Dodatkowe ikony.

Do projektu na najwyższą ocenę dołącz możliwość zapisu i odczytu danych z dołączonej wewnętrznej bazy danych. Utwórz klasę pomocniczą z połączeniem. Następnie utwórz i wykonaj zapytanie. Dane powinny wyświetlać się na pierwszej aktywności.

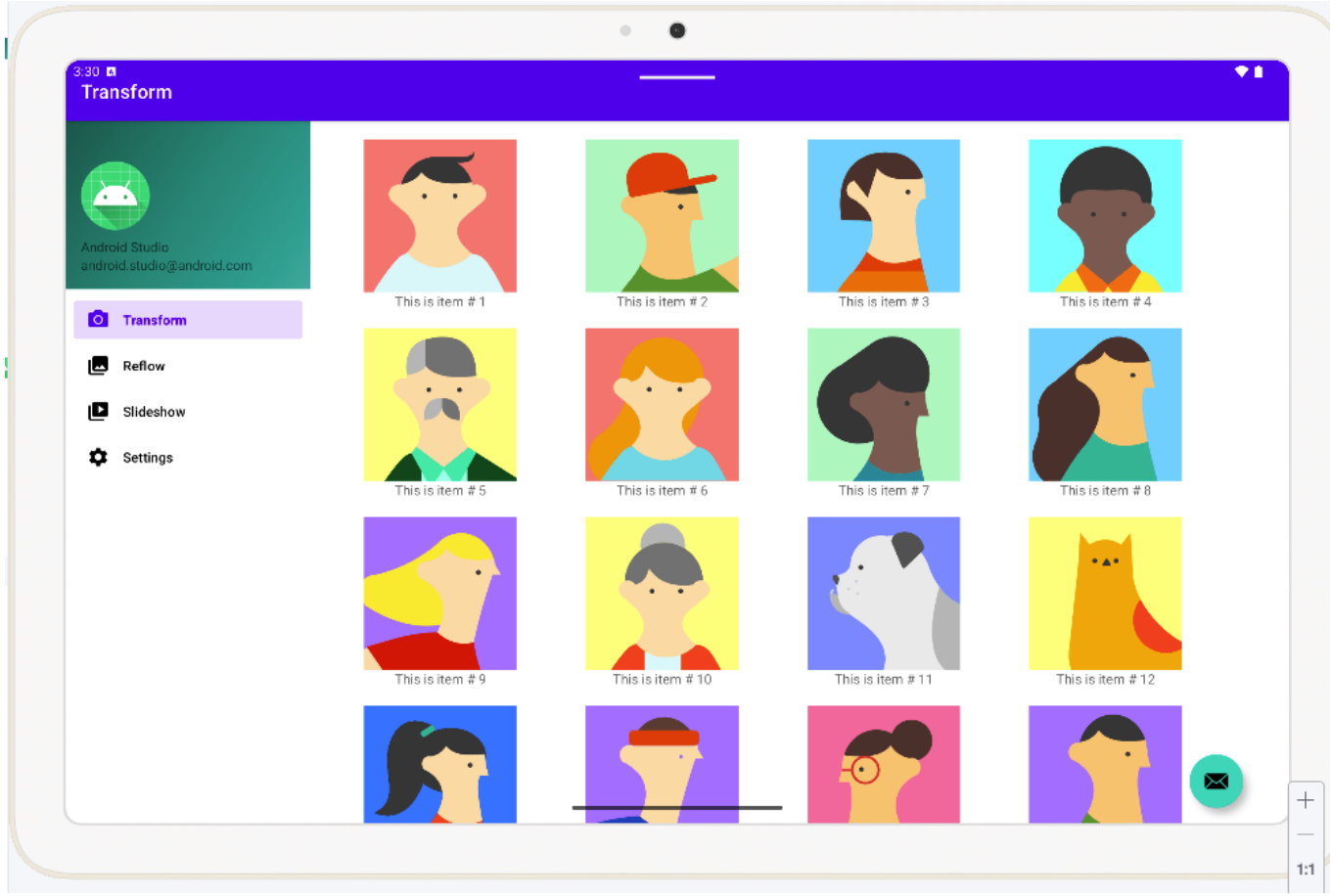
# Zadania do samodzielnego wykonania

## Zadanie 9 (na ocenę celującą)

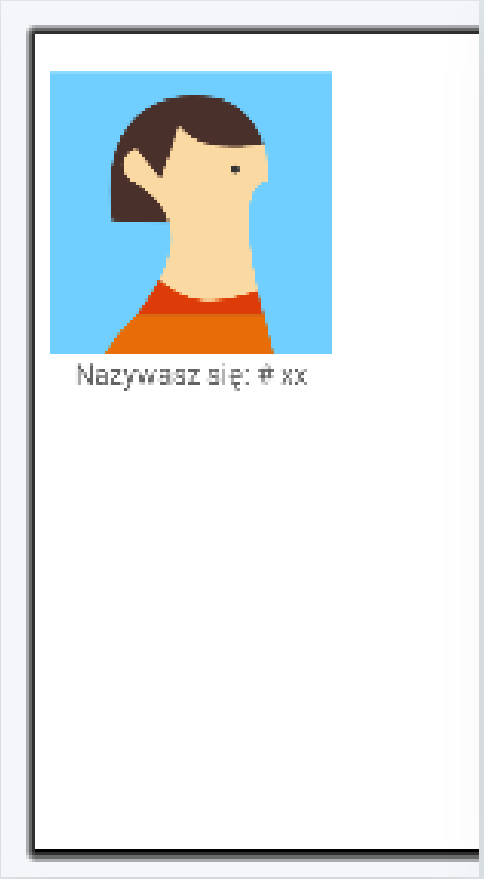
**Na podstawie rysunku przedstawiającego XML aplikacji stwórz aplikację „Moi znajomi”.**

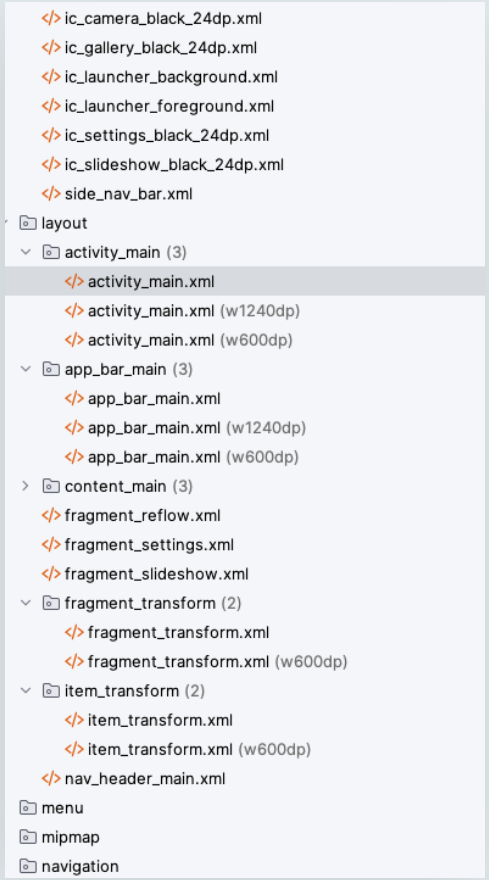
**Aplikacja zawiera:**

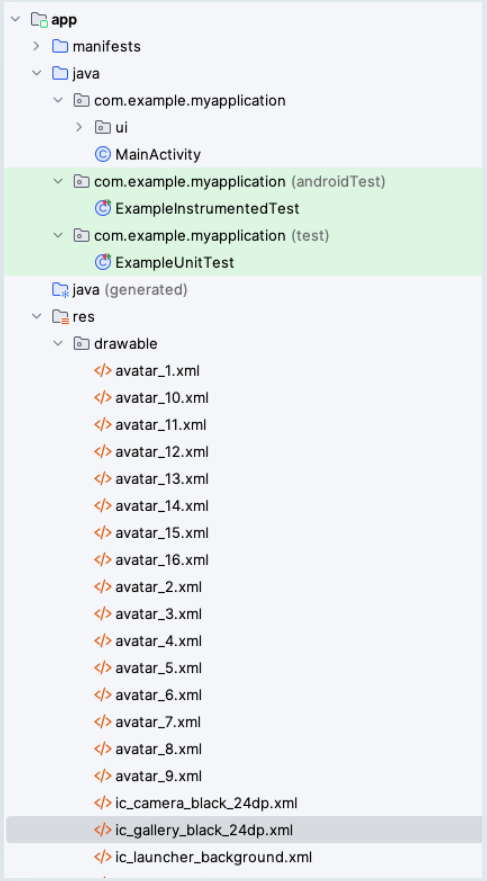
- **Wysuwany pasek z lewej strony z zalogowanym użytkownikiem oraz opcjami dodania zdjęcia, pokazu zdjęć oraz ustawień (tam znajdzie się informacja o aplikacji i wyjście)**
- **Karty z 16 zdjęciami oraz podpisem pod każdym z nich. Zamiast prawdziwych zdjęć można dodać te pochodzące z repozytorium Androida.**



# Zadania do samodzielnego wykonania




Działanie aplikacji:

- aplikacja umożliwia dostęp do zdjęć
  z pamięci telefonu oraz do galerii
- po naciśnięciu zdjęcia pojawia się okno z opisem osoby (np. płeć, wiek, zainteresowania).

- Dodaj przycisk do
- powracania do
- głównego ekranu.

# Zadania do samodzielnego wykonania

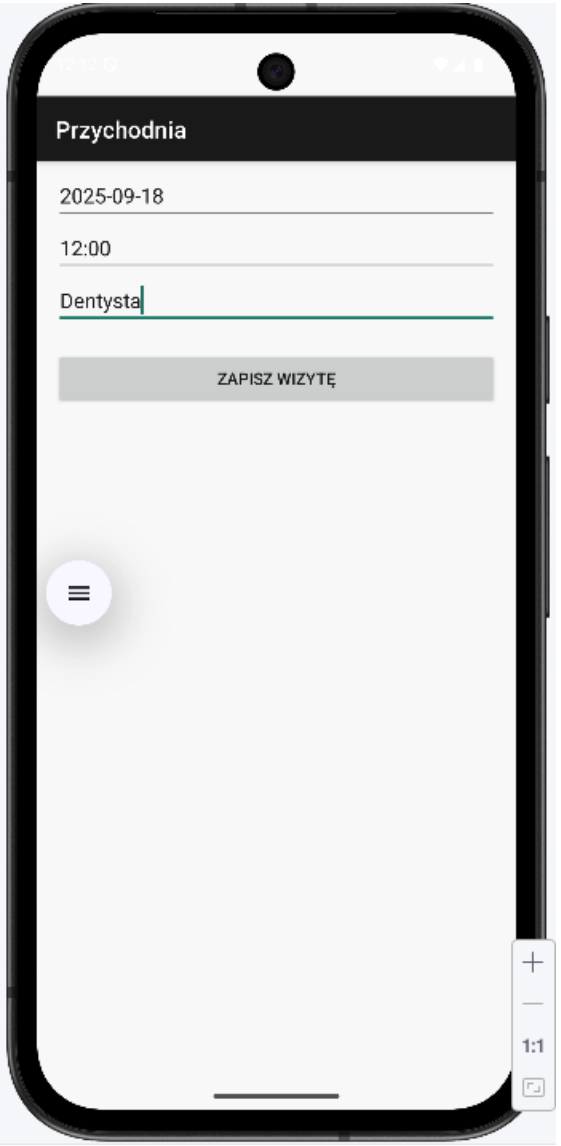
## Zadanie 10 (na 5 lub 6)

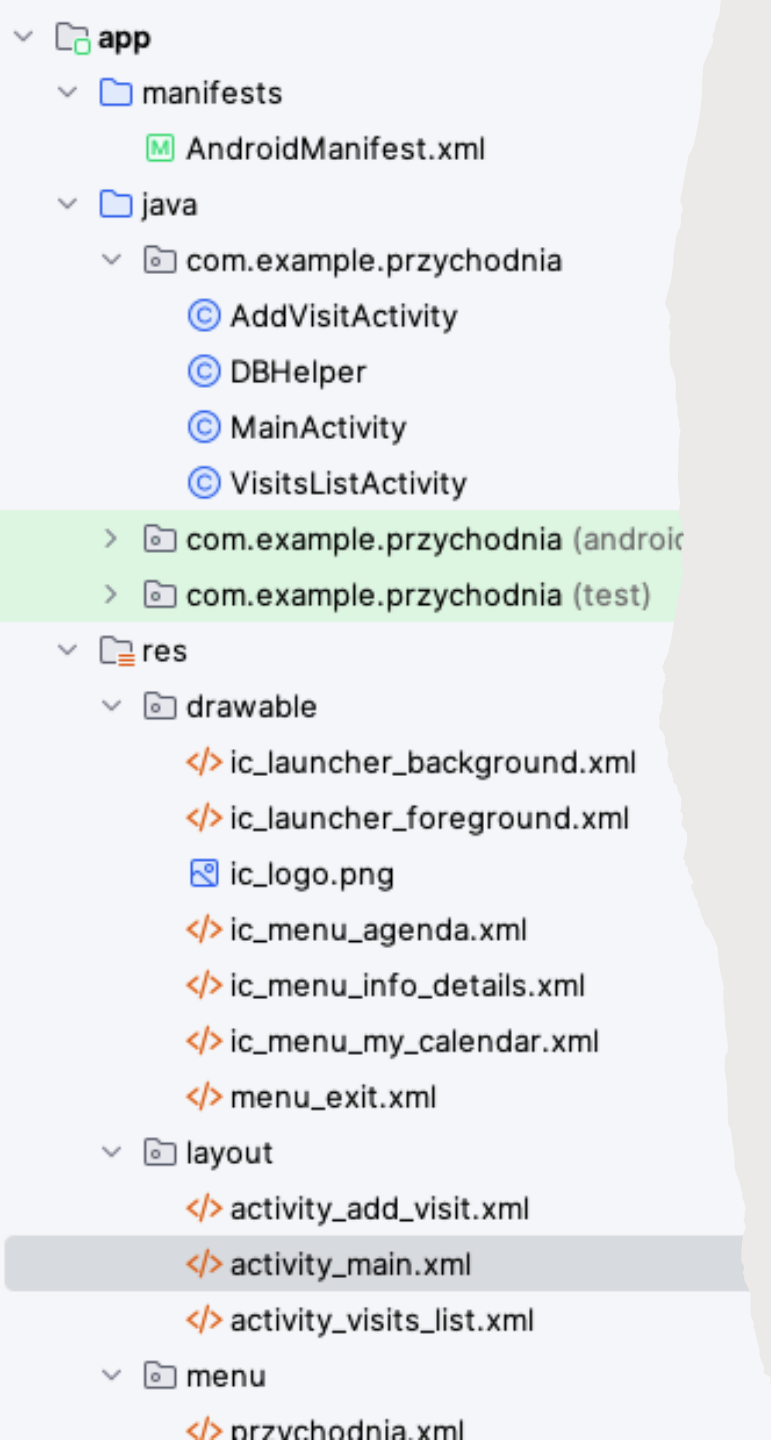
Na podstawie zdjęć zamieszczonych obok wykonaj aplikację do przychodni lekarskiej. Składa się z trzech Activity. Użytkownik zapisuje się na wizytę lekarską i może wyszukać terminy swoich wizyt.


# Zadania do samodzielnego wykonania

Aplikacja zapisuje dane w bazie SQLite. Użytkownik odczytuje dane za pomocą opcji w wysuwanym menu.

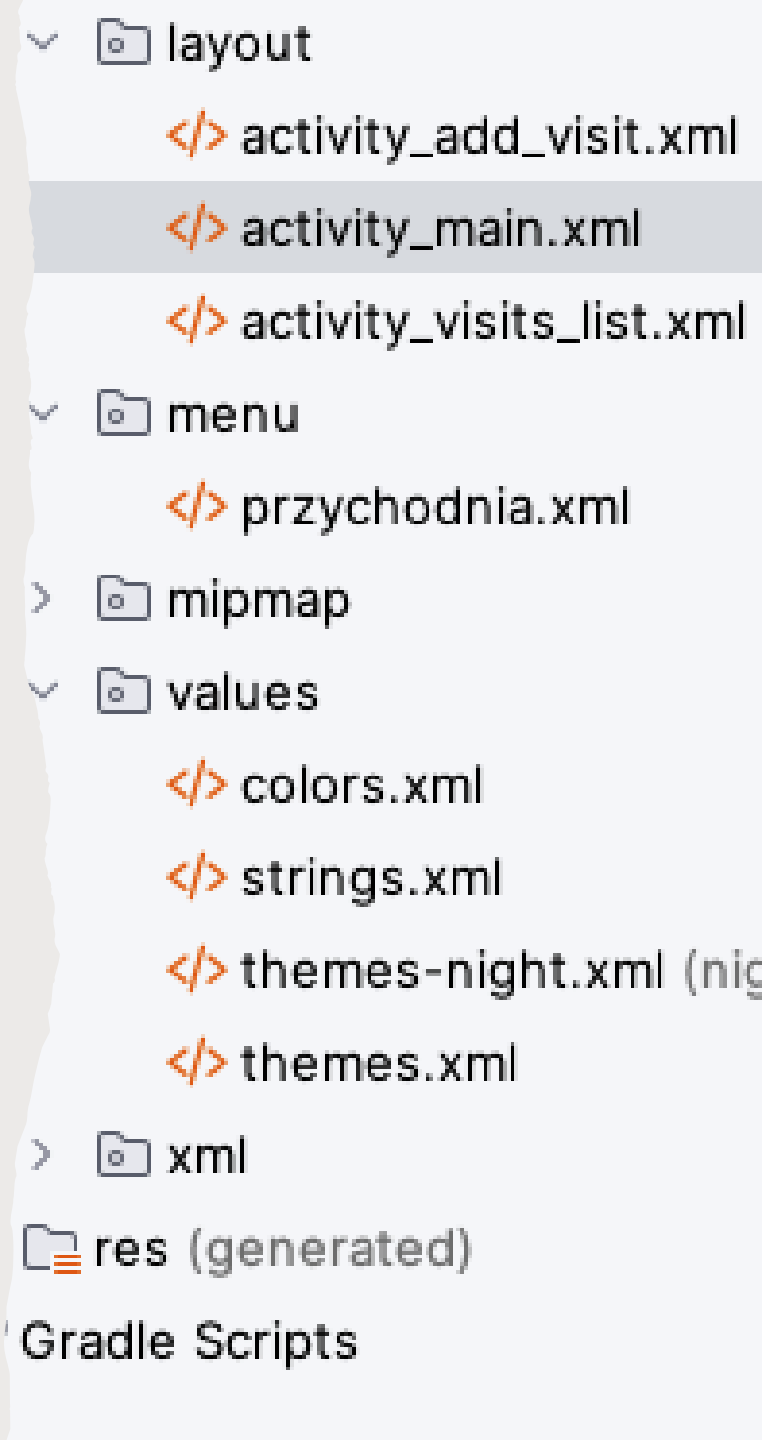
- Na najwyższą ocenę dodaj:
- Przyciski umożliwiające poruszanie się po każdej aktywności (mogą być strzałki)
- Zaproponuj i wykonaj estetyczny wygląd (dodaj ikony do menu)
- Dodaj możliwość przełączania się pomiędzy trybem jasnym i ciemnym, np. za pomocą ToggleButton jak na zdjęciu na dole.

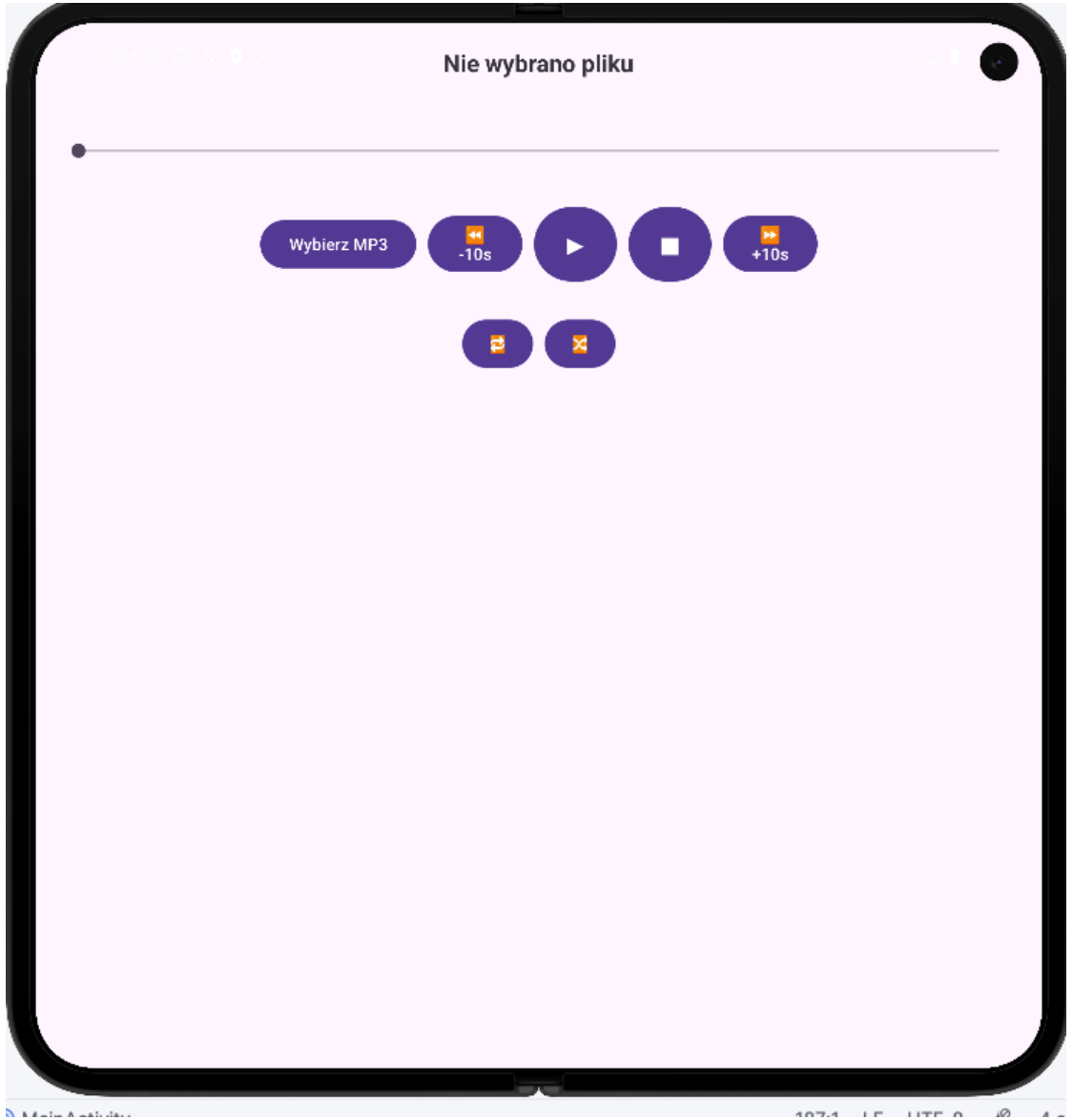

# Zadania do samodzielnego wykonania

Przykładowa struktura plików zawiera 3 Activity i klasę pomocniczą z połączeniem. Wysuwane menu ma własny layout. Zdefiniuj i dodaj do projektu własne kolory.



# Projekty multimedialne w Android Studio

Choć sam system zawiera już zainstalowane aplikacje do muzyki czy do filmów ... zawsze możemy spróbować napisać projekt, który spełni nasze oczekiwania.

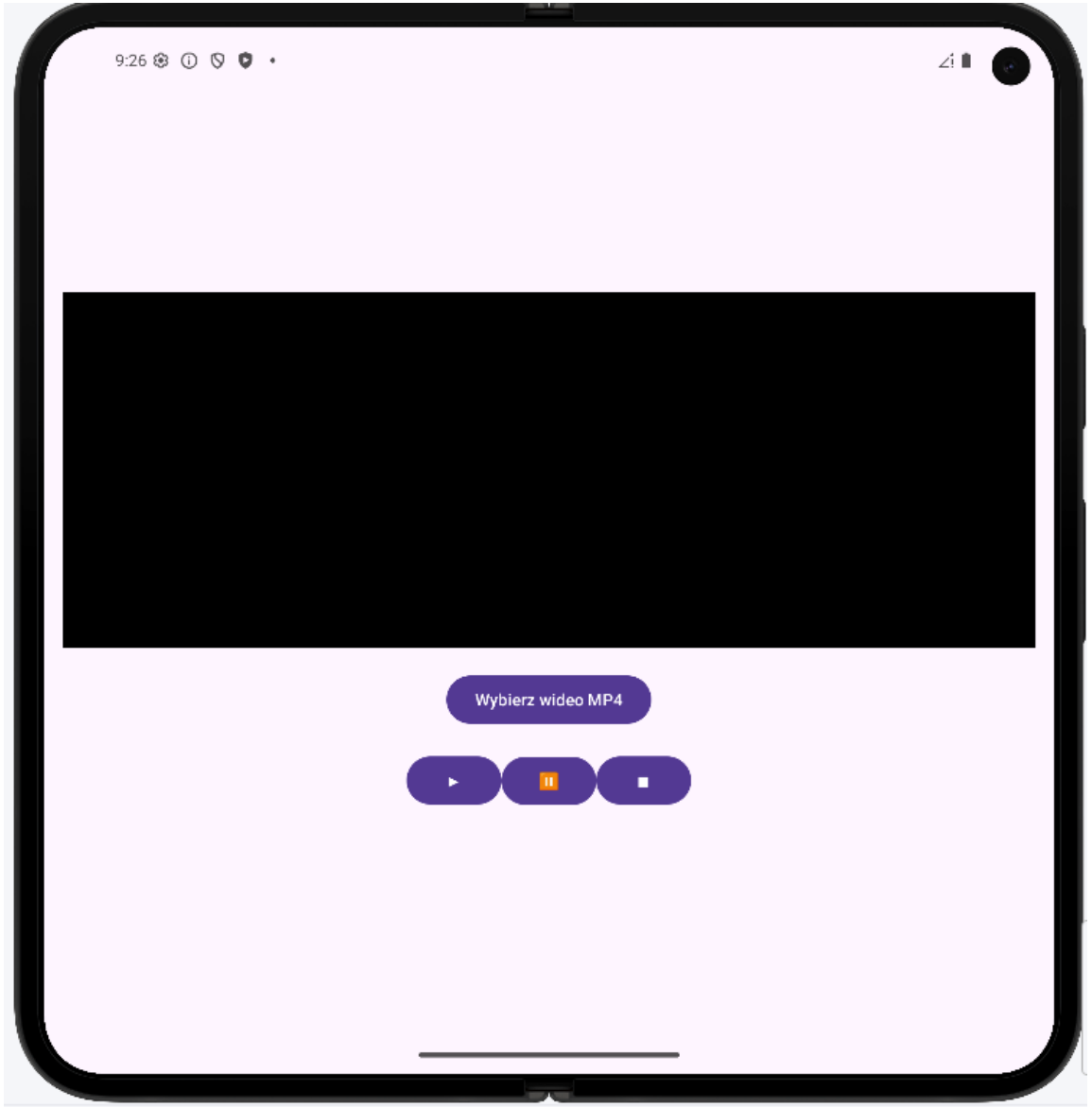


```
package com.example.mediaplayer;

import android.content.Intent;
import android.media.MediaMetadataRetriever;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.widget.Button;
import android.widget.SeekBar;
import android.widget.TextView;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private static final int PICK_AUDIO_REQUEST = 1;

    private TextView songTitle, songArtist;
    private SeekBar seekBar;
    private Button selectButton, playButton, stopButton,
forwardButton, backwardButton, repeatButton, shuffleButton;
    private MediaPlayer mediaPlayer;
    private Uri audioUri;

    private Handler handler = new Handler();
    private boolean isPlaying = false;
    private boolean isRepeat = false;

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

        songTitle = findViewById(R.id.songTitle);
        songArtist = findViewById(R.id.songArtist);
        seekBar = findViewById(R.id.seekBar);
        selectButton = findViewById(R.id.selectButton);
        playButton = findViewById(R.id.playButton);
        stopButton = findViewById(R.id.stopButton);
        forwardButton = findViewById(R.id.forwardButton);
        backwardButton = findViewById(R.id.backwardButton);
        repeatButton = findViewById(R.id.repeatButton);
        shuffleButton = findViewById(R.id.shuffleButton);

        selectButton.setOnClickListener(v -> selectAudioFile());

        playButton.setOnClickListener(v -> {
            if (mediaPlayer != null) {
                if (mediaPlayer.isPlaying()) {
                    mediaPlayer.pause();
                    playButton.setText("▶");
                    isPlaying = false;
                } else {
                    mediaPlayer.start();
                    playButton.setText("⏸");
                    isPlaying = true;
                    updateSeekBar();
                }
            }
        });

        stopButton.setOnClickListener(v -> {
            if (mediaPlayer != null) {
                mediaPlayer.stop();
                playButton.setText("▶");
                isPlaying = false;
                seekBar.setProgress(0);
                try {
                    mediaPlayer.prepare();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });

        forwardButton.setOnClickListener(v -> {
            if (mediaPlayer != null) {
                int pos = mediaPlayer.getCurrentPosition() + 10000; //
+10s
                if (pos > mediaPlayer.getDuration()) pos =
mediaPlayer.getDuration();
                mediaPlayer.seekTo(pos);
                seekBar.setProgress(pos);
            }
        });

        backwardButton.setOnClickListener(v -> {
            if (mediaPlayer != null) {
                int pos = mediaPlayer.getCurrentPosition() - 10000; //-
10s
                if (pos < 0) pos = 0;
                mediaPlayer.seekTo(pos);
                seekBar.setProgress(pos);
            }
        });

        repeatButton.setOnClickListener(v -> {
            isRepeat = !isRepeat;
            if (mediaPlayer != null) {
                mediaPlayer.setLooping(isRepeat);
            }
            repeatButton.setAlpha(isRepeat ? 1f : 0.5f);
        });

        shuffleButton.setOnClickListener(v -> {
            // Na razie tylko wizualny efekt (możesz rozbudować
playlistę i logikę)
            boolean active = shuffleButton.getAlpha() == 1f;
            shuffleButton.setAlpha(active ? 0.5f : 1f);
        });

        seekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
            public void onProgressChanged(SeekBar seekBar, int
progress, boolean fromUser) {
                if (mediaPlayer != null && fromUser) {
                    mediaPlayer.seekTo(progress);
                }
            }
            public void onStartTrackingTouch(SeekBar seekBar) {}
            public void onStopTrackingTouch(SeekBar seekBar) {}
        });
    }

    private void selectAudioFile() {
        Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
        intent.setType("audio/*");
        startActivityForResult(Intent.createChooser(intent,
"Wybierz plik audio"), PICK_AUDIO_REQUEST);
    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, @Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == PICK_AUDIO_REQUEST && resultCode
== RESULT_OK && data != null) {
            audioUri = data.getData();

            // Pobierz tytuł i artystę z metadanych
            MediaMetadataRetriever mmr = new
MediaMetadataRetriever();
            mmr.setDataSource(this, audioUri);

            String title =
mmr.extractMetadata(MediaMetadataRetriever.METADATA_KE
Y_TITLE);
            String artist =
mmr.extractMetadata(MediaMetadataRetriever.METADATA_KE
Y_ARTIST);
            if (title == null || title.isEmpty()) {
                title = getFileName(audioUri);
            }
            if (artist == null) artist = "";

            songTitle.setText(title);
            songArtist.setText(artist);

            if (mediaPlayer != null) {
                mediaPlayer.release();
            }

            mediaPlayer = MediaPlayer.create(this, audioUri);
            seekBar.setMax(mediaPlayer.getDuration());

            mediaPlayer.setOnCompletionListener(mp -> {
                playButton.setText("▶");
                isPlaying = false;
                seekBar.setProgress(0);
            });
        }
    }

    private void updateSeekBar() {
        if (mediaPlayer != null) {
            seekBar.setProgress(mediaPlayer.getCurrentPosition());
            if (isPlaying) {
                handler.postDelayed(this::updateSeekBar, 500);
            }
        }
    }

    private String getFileName(Uri uri) {
        String path = uri.getPath();
        if (path != null && path.contains("/")) {
            return path.substring(path.lastIndexOf("/") + 1);
        }
        return "plik audio";
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (mediaPlayer != null) {
            mediaPlayer.release();
        }
    }
}
```

# Odtwarzacz MP3

```
<uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:padding="20dp"
  android:gravity="center_horizontal"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <TextView
    android:id="@+id/songTitle"
    android:text="Nie wybrano pliku"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_marginBottom="10dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

  <TextView
    android:id="@+id/songArtist"
    android:text=""
    android:textSize="16sp"
    android:textColor="#555555"
    android:layout_marginBottom="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

  <SeekBar
    android:id="@+id/seekBar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

  <LinearLayout
    android:orientation="horizontal"
    android:layout_marginTop="30dp"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
      android:id="@+id/selectButton"
      android:text="Wybierz MP3"
      android:layout_margin="5dp"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />

    <Button
      android:id="@+id/backwardButton"
      android:text="⏪ -10s"
      android:layout_margin="5dp"
      android:layout_width="80dp"
      android:layout_height="wrap_content" />

    <Button
      android:id="@+id/playButton"
      android:text="▶"
      android:textSize="24sp"
      android:layout_margin="5dp"
      android:layout_width="70dp"
      android:layout_height="70dp" />

    <Button
      android:id="@+id/stopButton"
      android:text="■"
      android:textSize="24sp"
      android:layout_margin="5dp"
      android:layout_width="70dp"
      android:layout_height="70dp" />

    <Button
      android:id="@+id/forwardButton"
      android:text="⏩ +10s"
      android:layout_margin="5dp"
      android:layout_width="80dp"
      android:layout_height="wrap_content" />
  </LinearLayout>

  <LinearLayout
    android:orientation="horizontal"
    android:gravity="center"
    android:layout_marginTop="15dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
      android:id="@+id/repeatButton"
      android:text="🔁"
      android:layout_margin="5dp"
      android:layout_width="60dp"
      android:layout_height="wrap_content" />

    <Button
      android:id="@+id/shuffleButton"
      android:text="🔀"
      android:layout_margin="5dp"
      android:layout_width="60dp"
      android:layout_height="wrap_content" />
  </LinearLayout>
</LinearLayout>
```



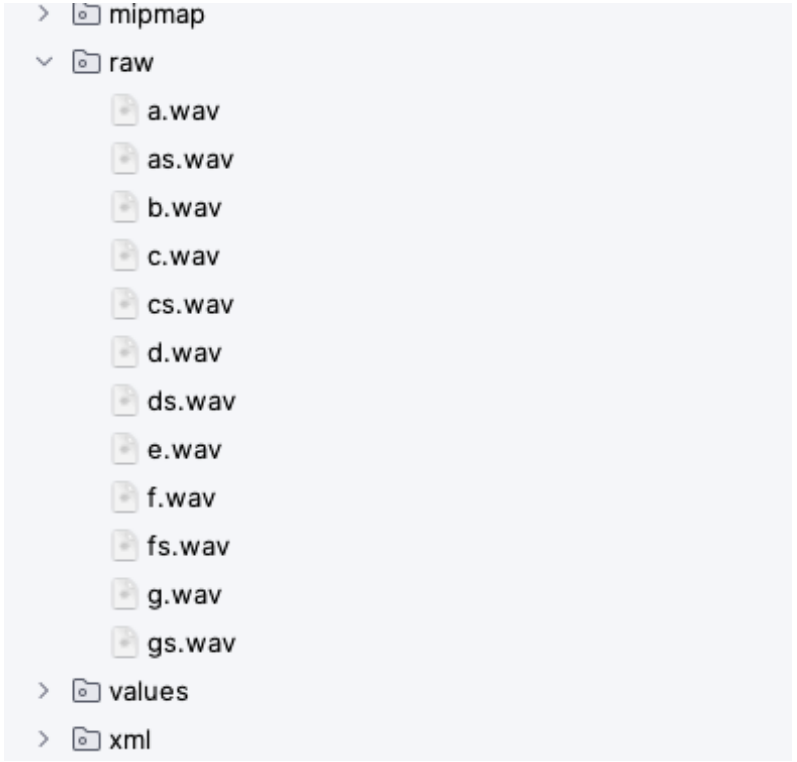

# Odtwarzacz MP4

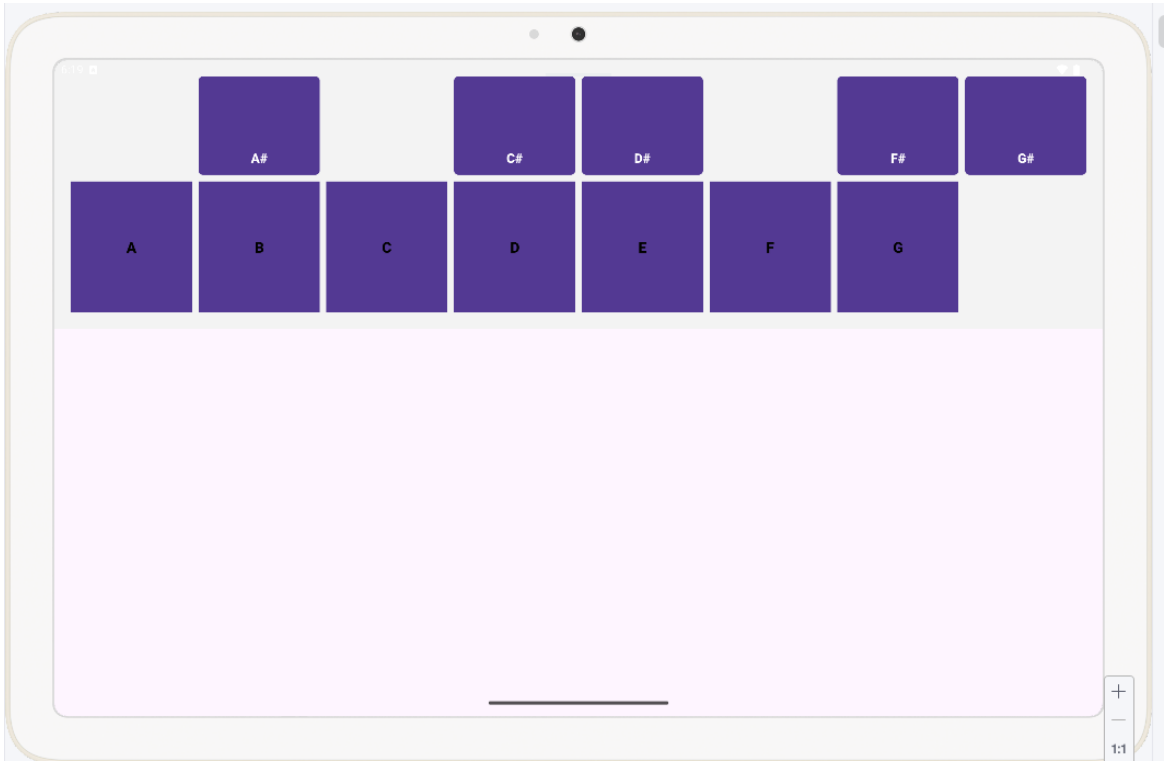
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/
apk/res/android"
  android:orientation="vertical"
  android:gravity="center"
  android:padding="16dp"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <VideoView
    android:id="@+id/videoView"
    android:layout_width="match_parent"
    android:layout_height="300dp"
    android:layout_marginBottom="20dp" />

  <Button
    android:id="@+id/selectVideoButton"
    android:text="Wybierz wideo MP4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

  <LinearLayout
    android:orientation="horizontal"
    android:layout_marginTop="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <Button
      android:id="@+id/playButton"
      android:text="▶ "
      android:layout_width="80dp"
      android:layout_height="wrap_content" />

    <Button
      android:id="@+id/pauseButton"
      android:text="⏸ "
      android:layout_width="80dp"
      android:layout_height="wrap_content" />

    <Button
      android:id="@+id/stopButton"
      android:text="■"
      android:layout_width="80dp"
      android:layout_height="wrap_content" />
  </LinearLayout>
</LinearLayout>
```

# Odtwarzacz MP4

```
package com.example.mp4player;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.widget.Button;
import android.widget.VideoView;
import android.widget.MediaController;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private static final int PICK_VIDEO_REQUEST = 1;
    private VideoView videoView;
    private Button selectVideoButton, playButton, pauseButton,
stopButton;
    private Uri videoUri;

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

        videoView = findViewById(R.id.videoView);
        selectVideoButton = findViewById(R.id.selectVideoButton);
        playButton = findViewById(R.id.playButton);
        pauseButton = findViewById(R.id.pauseButton);
        stopButton = findViewById(R.id.stopButton);

        // Dodaj kontrolki odtwarzacza do VideoView (play/pause/seek)
        MediaController mediaController = new MediaController(this);
        mediaController.setAnchorView(videoView);
        videoView.setMediaController(mediaController);

        selectVideoButton.setOnClickListener(v -> {
            Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
            intent.setType("video/mp4");
            startActivityForResult(Intent.createChooser(intent, "Wybierz
wideo MP4"), PICK_VIDEO_REQUEST);
        });

        playButton.setOnClickListener(v -> {
            if (videoUri != null && !videoView.isPlaying()) {
                videoView.start();
            }
        });

        pauseButton.setOnClickListener(v -> {
            if (videoView.isPlaying()) {
                videoView.pause();
            }
        });

        stopButton.setOnClickListener(v -> {
            if (videoView.isPlaying()) {
                videoView.stopPlayback();
                // Po zatrzymaniu, aby odtworzyć ponownie, trzeba ustawić
źródło na nowo
                videoView.setVideoURI(videoUri);
            }
        });
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode,
@Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == PICK_VIDEO_REQUEST && resultCode ==
RESULT_OK && data != null) {
            videoUri = data.getData();
            videoView.setVideoURI(videoUri);
        }
    }
}
```

# „Cymbałki”


mipmap
raw
a.wav
as.wav
b.wav
c.wav
cs.wav
d.wav
ds.wav
e.wav
f.wav
fs.wav
g.wav
gs.wav
values
xml

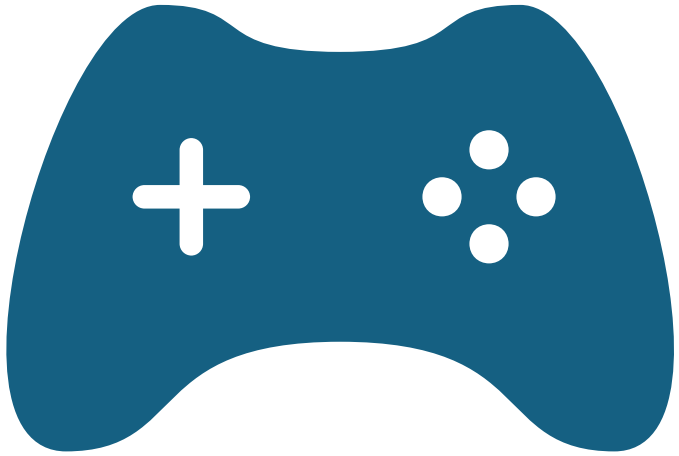

A#
C#
D#
F#
G#
A
B
C
D
E
F
G
1:1

# „Cymbałki”

```
package com.example.piannino;

import android.media.AudioManager;
import android.media.SoundPool;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private SoundPool soundPool;

    // deklaracje dla wszystkich dźwięków
    private int soundA, soundAs, soundB, soundC, soundCs, soundD,
soundDs, soundE, soundF, soundFs, soundG, soundGs;

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

        soundPool = new SoundPool(12, AudioManager.STREAM_MUSIC, 0);

        // ładujemy pliki do zmiennych
        soundA = soundPool.load(this, R.raw.a, 1);
        soundAs = soundPool.load(this, R.raw.as, 1);
        soundB = soundPool.load(this, R.raw.b, 1);
        soundC = soundPool.load(this, R.raw.c, 1);
        soundCs = soundPool.load(this, R.raw.cs, 1);
        soundD = soundPool.load(this, R.raw.d, 1);
        soundDs = soundPool.load(this, R.raw.ds, 1);
        soundE = soundPool.load(this, R.raw.e, 1);
        soundF = soundPool.load(this, R.raw.f, 1);
        soundFs = soundPool.load(this, R.raw.fs, 1);
        soundG = soundPool.load(this, R.raw.g, 1);
        soundGs = soundPool.load(this, R.raw.gs, 1);

        Button keyA = findViewById(R.id.a);
        Button keyAs = findViewById(R.id.as);
        Button keyB = findViewById(R.id.b);
        Button keyC = findViewById(R.id.c);
        Button keyCs = findViewById(R.id.cs);
        Button keyD = findViewById(R.id.d);
        Button keyDs = findViewById(R.id.ds);
        Button keyE = findViewById(R.id.e);
        Button keyF = findViewById(R.id.f);
        Button keyFs = findViewById(R.id.fs);
        Button keyG = findViewById(R.id.g);
        Button keyGs = findViewById(R.id.gs);

        keyA.setOnClickListener(v -> soundPool.play(soundA, 1, 1, 0, 0, 1));
        keyAs.setOnClickListener(v -> soundPool.play(soundAs, 1, 1, 0, 0, 1));
        keyB.setOnClickListener(v -> soundPool.play(soundB, 1, 1, 0, 0, 1));
        keyC.setOnClickListener(v -> soundPool.play(soundC, 1, 1, 0, 0, 1));
        keyCs.setOnClickListener(v -> soundPool.play(soundCs, 1, 1, 0, 0,
1));
        keyD.setOnClickListener(v -> soundPool.play(soundD, 1, 1, 0, 0, 1));
        keyDs.setOnClickListener(v -> soundPool.play(soundDs, 1, 1, 0, 0,
1));
        keyE.setOnClickListener(v -> soundPool.play(soundE, 1, 1, 0, 0, 1));
        keyF.setOnClickListener(v -> soundPool.play(soundF, 1, 1, 0, 0, 1));
        keyFs.setOnClickListener(v -> soundPool.play(soundFs, 1, 1, 0, 0, 1));
        keyG.setOnClickListener(v -> soundPool.play(soundG, 1, 1, 0, 0, 1));
        keyGs.setOnClickListener(v -> soundPool.play(soundGs, 1, 1, 0, 0,
1));
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        soundPool.release();
        soundPool = null;
    }
}
```

MainActivity

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gridLayout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:columnCount="8"
    android:rowCount="2"
    android:padding="16dp"
    android:background="#f5f5f5"
    android:alignmentMode="alignMargins"
    android:useDefaultMargins="true">

    <!-- Rząd 0 - czarne klawisze (#) -->

    <View
        android:layout_width="0dp"
        android:layout_height="80dp"
        android:layout_row="0"
        android:layout_column="0"
        android:layout_columnWeight="1" />

    <Button
        android:id="@+id/as"
        android:layout_width="0dp"
        android:layout_height="120dp"
        android:layout_row="0"
        android:layout_column="1"
        android:layout_columnWeight="1"

android:background="@drawable/black_key_border"
        android:text="A#"
        android:textColor="#FFF"
        android:textSize="16sp"
        android:textStyle="bold"
        android:gravity="bottom|center_horizontal"
        android:elevation="6dp" />

    <View
        android:layout_width="0dp"
        android:layout_height="80dp"
        android:layout_row="0"
        android:layout_column="2"
        android:layout_columnWeight="1" />

    <Button
        android:id="@+id/cs"
        android:layout_width="0dp"
        android:layout_height="120dp"
        android:layout_row="0"
        android:layout_column="3"
        android:layout_columnWeight="1"

android:background="@drawable/black_key_border"
        android:text="C#"
        android:textColor="#FFF"
        android:textSize="16sp"
        android:textStyle="bold"
        android:gravity="bottom|center_horizontal"
        android:elevation="6dp" />

    <Button
        android:id="@+id/ds"
        android:layout_width="0dp"
        android:layout_height="120dp"
        android:layout_row="0"
        android:layout_column="4"
        android:layout_columnWeight="1"

android:background="@drawable/black_key_border"
        android:text="D#"
        android:textColor="#FFF"
        android:textSize="16sp"
        android:textStyle="bold"
        android:gravity="bottom|center_horizontal"
        android:elevation="6dp" />

    <View
        android:layout_width="0dp"
        android:layout_height="80dp"
        android:layout_row="0"
        android:layout_column="5"
        android:layout_columnWeight="1" />

    <Button
        android:id="@+id/fs"
        android:layout_width="0dp"
        android:layout_height="120dp"
        android:layout_row="0"
        android:layout_column="6"
        android:layout_columnWeight="1"

android:background="@drawable/black_key_border"
        android:text="F#"
        android:textColor="#FFF"
        android:textSize="16sp"
        android:textStyle="bold"
        android:gravity="bottom|center_horizontal"
        android:elevation="6dp" />

    <Button
        android:id="@+id/gs"
        android:layout_width="0dp"
        android:layout_height="120dp"
        android:layout_row="0"
        android:layout_column="7"
        android:layout_columnWeight="1"

android:background="@drawable/black_key_border"
        android:text="G#"
        android:textColor="#FFF"
        android:textSize="16sp"
        android:textStyle="bold"
        android:gravity="bottom|center_horizontal"
        android:elevation="6dp" />

    <!-- Rząd 1 - białe klawisze -->

    <Button
        android:id="@+id/a"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="0"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="A"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/b"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="1"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="B"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/c"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="2"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="C"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/d"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="3"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="D"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/e"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="4"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="E"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/f"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="5"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="F"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/g"
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="6"
        android:layout_columnWeight="1"

android:background="@drawable/white_key_border"
        android:text="G"
        android:textColor="#000"
        android:textSize="18sp"
        android:textStyle="bold" />

    <View
        android:layout_width="0dp"
        android:layout_height="160dp"
        android:layout_row="1"
        android:layout_column="7"
        android:layout_columnWeight="1" />

</GridLayout>
```

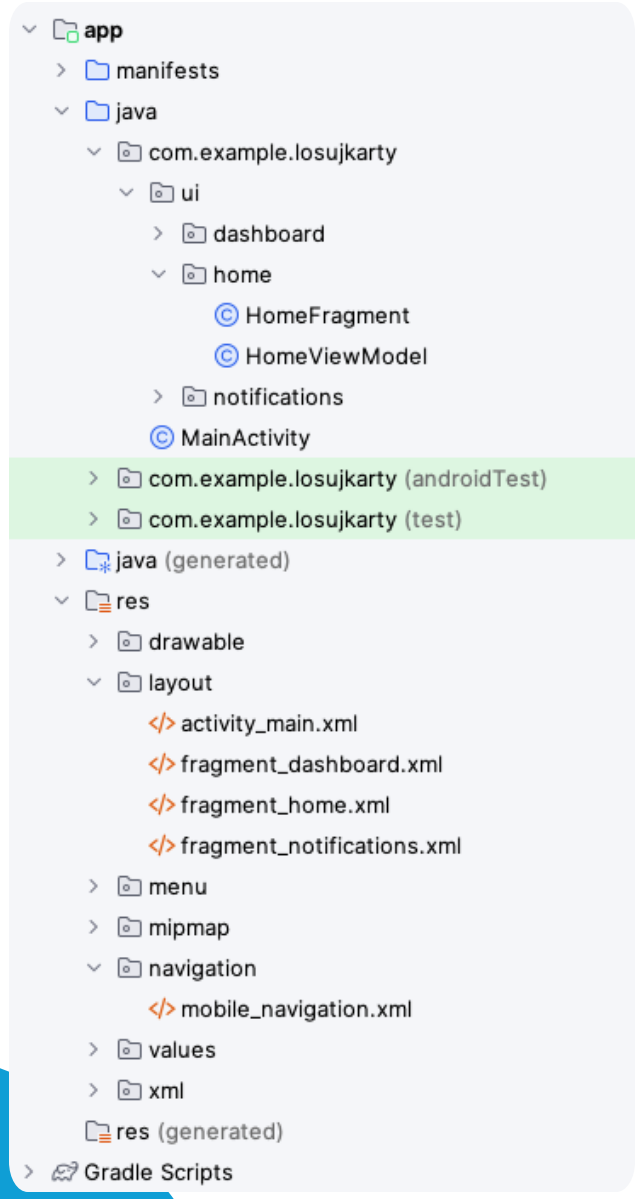


# Gry w Android Studio

- Gry w Androidzie tworzymy jako aplikacje z interaktywnym GUI.
- Możliwe podejścia:
- **Canvas i SurfaceView** – rysowanie grafik i animacji.
- **OpenGL ES** – zaawansowana grafika 3D.
- Silniki gier (np. Unity, libGDX) – ale tu skupimy się na natywnym podejściu w Javie.

# Przykład gry

Użytkownik wybiera kartę i naciska przycisk. Po czym karta zostaje wylosowana z puli dostępnych widoków. Jeżeli użytkownik trafi to wyświetla się komunikat. Prawda, że proste?




# Schemat i kod gry

```
package com.example.losujkarty;

import android.os.Bundle;
import android.view.View;
import android.widget.*;
import androidx.appcompat.app.AppCompatActivity;
import java.util.HashMap;
import java.util.Random;

public class MainActivity extends AppCompatActivity {

    private ImageView imgUserCard, imgDrawnCard;
    private TextView txtResult;
    private Spinner spinnerCards;
    private Button btnDraw;

    private String[] cardNames = {"as_pik", "krol_kier", "dama_karo", "walet_trefl"};
    private HashMap<String, Integer> cardImages;

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

        spinnerCards = findViewById(R.id.spinnerCards);
        btnDraw = findViewById(R.id.btnDraw);
        imgUserCard = findViewById(R.id.imgUserCard);
        imgDrawnCard = findViewById(R.id.imgDrawnCard);
        txtResult = findViewById(R.id.txtResult);

        // Mapa z obrazami kart
        cardImages = new HashMap<>();
        cardImages.put("as_pik", R.drawable.as_pik);
        cardImages.put("krol_kier", R.drawable.krol_kier);
        cardImages.put("dama_karo", R.drawable.dama_karo);
        cardImages.put("walet_trefl", R.drawable.walet_trefl);

        // Spinner - wybór kart
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_dropdown_item, cardNames);
        spinnerCards.setAdapter(adapter);

        btnDraw.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                drawCard();
            }
        });
    }

    private void drawCard() {
        String selectedCard = spinnerCards.getSelectedItem().toString();
        imgUserCard.setImageResource(cardImages.get(selectedCard));

        // Losowanie karty
        Random random = new Random();
        String drawnCard = cardNames[random.nextInt(cardNames.length)];
        imgDrawnCard.setImageResource(cardImages.get(drawnCard));

        // Porównanie kart
        if (selectedCard.equals(drawnCard)) {
            txtResult.setText("Gratulacje! Trafiłeś swoją kartę!");
        } else {
            txtResult.setText("Spróbuj ponownie! Wylosowana karta: " + drawnCard);
        }
    }
}
```

# Kod gry

```
package com.example.losujkarty.ui.home;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;

import com.example.losujkarty.databinding.FragmentHomeBinding;

public class HomeFragment extends Fragment {

    private FragmentHomeBinding binding;

    private HomeFragment() {
    }

    public static HomeFragment createHomeFragment() {
        return new HomeFragment();
    }

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        HomeViewModel homeViewModel =
                new ViewModelProvider(this).get(HomeViewModel.class);

        binding = FragmentHomeBinding.inflate(inflater, container, false);
        View root = binding.getRoot();

        final TextView textView = binding.textHome;
        homeViewModel.getText().observe(getViewLifecycleOwner(), textView::setText);
        return root;
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }
}
```

FRAGMENT

```
package com.example.losujkarty.ui.home;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;

import
com.example.losujkarty.databinding.FragmentHomeBinding;

public class HomeFragment extends Fragment {

    private FragmentHomeBinding binding;

    private HomeFragment() {
    }

    public static HomeFragment createHomeFragment() {
        return new HomeFragment();
    }

    public View onCreateView(@NonNull LayoutInflater
inflater,
                             ViewGroup container, Bundle
savedInstanceState) {
        HomeViewModel homeViewModel =
                new
ViewModelProvider(this).get(HomeViewModel.class);

        binding = FragmentHomeBinding.inflate(inflater,
container, false);
        View root = binding.getRoot();

        final TextView textView = binding.textHome;

homeViewModel.getText().observe(getViewLifecycleOwner(),
textView::setText);
        return root;
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }
}
```

View Model – służy do przechowywania danych ukrytych we Fragmentach.

# Główne elementy gier w JAVA

### Główne elementy Androida w grach:

**Activity** — pojedynczy ekran aplikacji, odpowiada za interfejs i logikę.

**View** — elementy UI do wyświetlania i interakcji (np. przyciski, pola tekstowe).

**SurfaceView** — specjalny widok umożliwiający rysowanie grafiki w oddzielnym wątku, ważny dla płynnej animacji.

### Rysowanie i animacja:

W prostych grach można rysować na Canvas w metodzie onDraw().

W bardziej wymagających grach używamy SurfaceView lub TextureView, aby oddzielić rendering od głównego wątku UI.

### Eventy dotykowe i sterowanie:

Obsługa kliknięć, przeciągnięć i gestów to podstawa interakcji w grach mobilnych.

# Rozszerzenia i rozwój gry

- **Timer** – ograniczenie czasu gry.
  Możemy użyć klasy CountDownTimer, która odlicza czas i po zakończeniu wyświetla wynik.
- **Przechowywanie wyników** – zapisywanie najlepszych wyników w SharedPreferences, by były dostępne między uruchomieniami aplikacji.
- **Animacje i dźwięki** – Animacje można robić za pomocą klasy Animation lub rysując na Canvas zmieniające się obiekty. Dźwięki (kliknięcia, efekty) dodaje się przez SoundPool lub MediaPlayer.
- **Więcej elementów gry** – np. przesuwające się przeszkody, punkty do zbierania, liczniki życia.
- **Menu i ekran końcowy** – dodatkowe Activity lub Fragmenty do wyboru poziomu, pauzy i podsumowania wyników.

# Dobre praktyki

## Wydajność

Unikaj wykonywania ciężkich operacji w głównym wątku UI (np. długie pętle lub zapytania sieciowe).

Używaj wątków, AsyncTask lub Handler.

## Obsługa różnych rozdzielczości

Używaj elastycznych układów (Layout), wymiarów w dp i zasobów obrazów w różnych rozdzielczościach.

## Testowanie

Testuj na różnych urządzeniach i wersjach Androida.

Warto testować zachowanie podczas zmiany orientacji ekranu.

## Czytelność kodu

Rozdziel logikę gry od UI, np. przez klasy modelu i kontrolery.

Komentuj kod i stosuj wzorce projektowe.

# Narzędzia i biblioteki

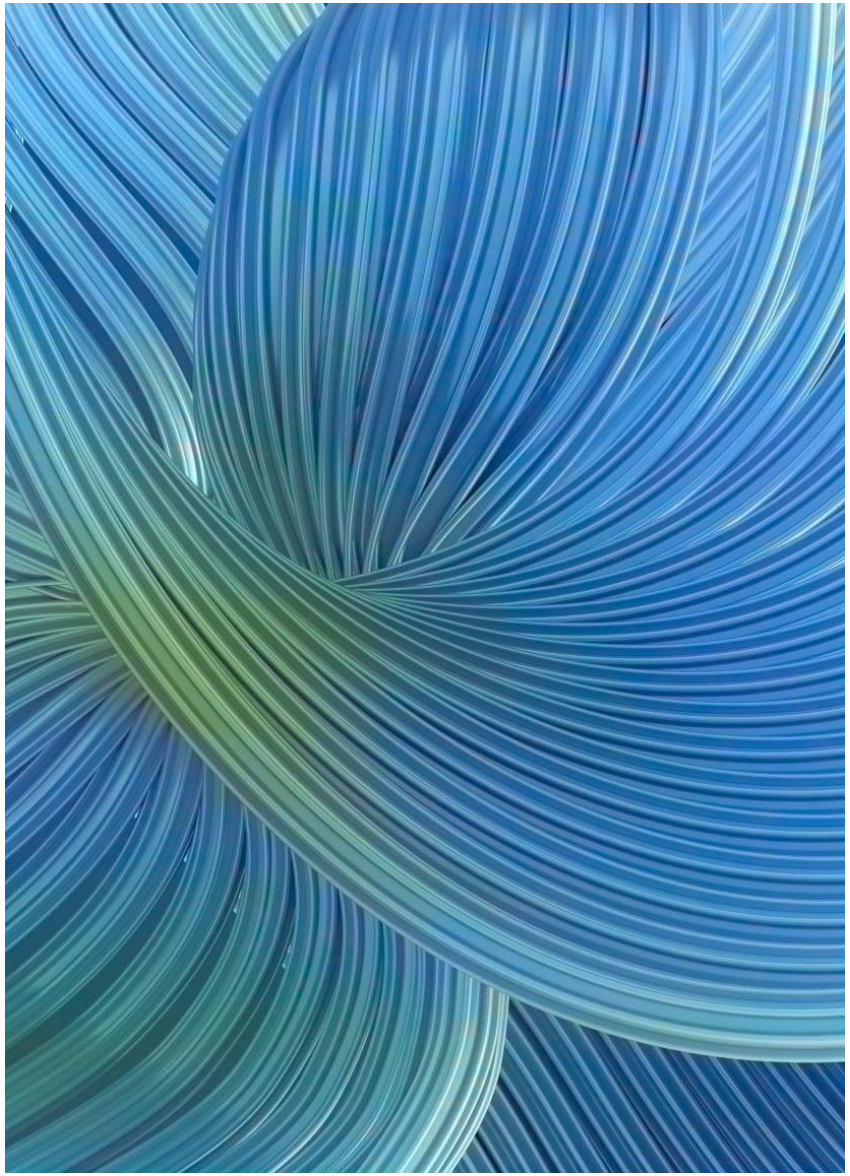
**libGDX** – popularny framework Java do gier 2D i 3D, działający na Androidzie, PC i iOS.

**AndEngine** – prosty silnik 2D dla Androida.

**Unity** lub **Unreal Engine** – bardziej rozbudowane silniki, ale wymagają znajomości C# lub C++.

**Grafika i zasoby**

- Programy do tworzenia sprite'ów: GIMP, Photoshop, Aseprite.
- Dźwięki: Audacity, Bfxr.



# Zastosowanie Canvas

```
private void draw() {
    if (holder.getSurface().isValid()) {
        Canvas canvas = holder.lockCanvas();
        canvas.drawColor(Color.BLACK); // Tło
        paint.setColor(Color.GREEN);
        paint.setTextSize(50);
        canvas.drawText("Gra w Android Studio!", 50, 100, paint);
        canvas.drawCircle(x, 300, 50, paint); // Ruchome koło
        holder.unlockCanvasAndPost(canvas);
    }
}
```

# Ruch postaci

```
public boolean onTouchEvent(MotionEvent event) {
   float tx = event.getX();
   float ty = event.getY();

   switch (event.getAction()) {
      case MotionEvent.ACTION_DOWN:
         // Ruch w zależności od strefy dotyku
         if (tx < getWidth() / 3f) {
            dx = -10; dy = 0;
         } else if (tx > 2 * getWidth() / 3f) {
            dx = 10; dy = 0;
         } else if (ty < getHeight() / 2f) {
            dy = -10; dx = 0;
         } else {
            dy = 10; dx = 0;
         }
         break;
   }

   return true;
}
```

# Sterowanie ruchem postaci

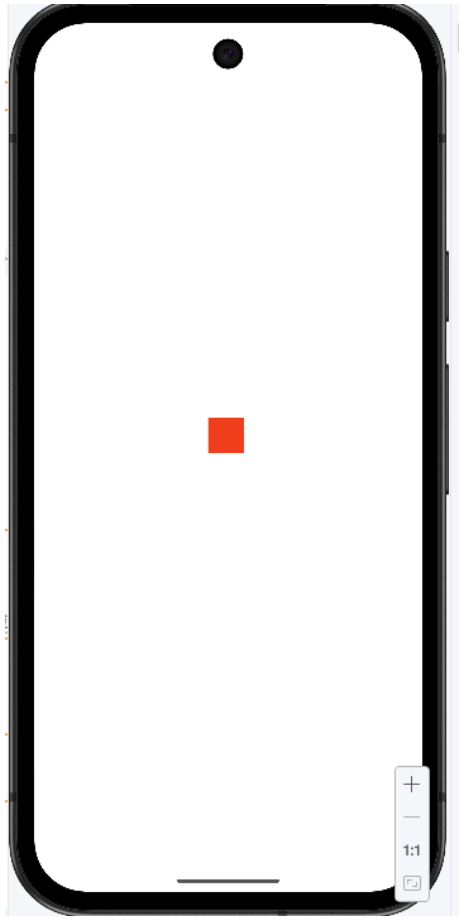
```
@Override
public boolean onTouchEvent(MotionEvent event) {
    int action = event.getAction();
    float touchX = event.getX();
    float touchY = event.getY();

    switch (action) {
        case MotionEvent.ACTION_DOWN:
        case MotionEvent.ACTION_MOVE:
            // Ustaw współrzędne postaci na pozycję palca
            characterX = touchX;
            characterY = touchY;
            break;
        case MotionEvent.ACTION_UP:
            // Możesz zatrzymać ruch lub inne akcje
            break;
    }
    return true; // obsłużyliśmy zdarzenie
}
```

Ruch palca

```
@Override
public boolean onKeyDown(int
keyCode, KeyEvent event) {
    switch (keyCode) {
        case
KeyEvent.KEYCODE_DPAD_UP:
            characterY -= 10;
            return true;
        case
KeyEvent.KEYCODE_DPAD_DOWN:
            characterY += 10;
            return true;
        case
KeyEvent.KEYCODE_DPAD_LEFT:
            characterX -= 10;
            return true;
        case
KeyEvent.KEYCODE_DPAD_RIGHT:
            characterX += 10;
            return true;
    }
    return super.onKeyDown(keyCode,
event);
}
```

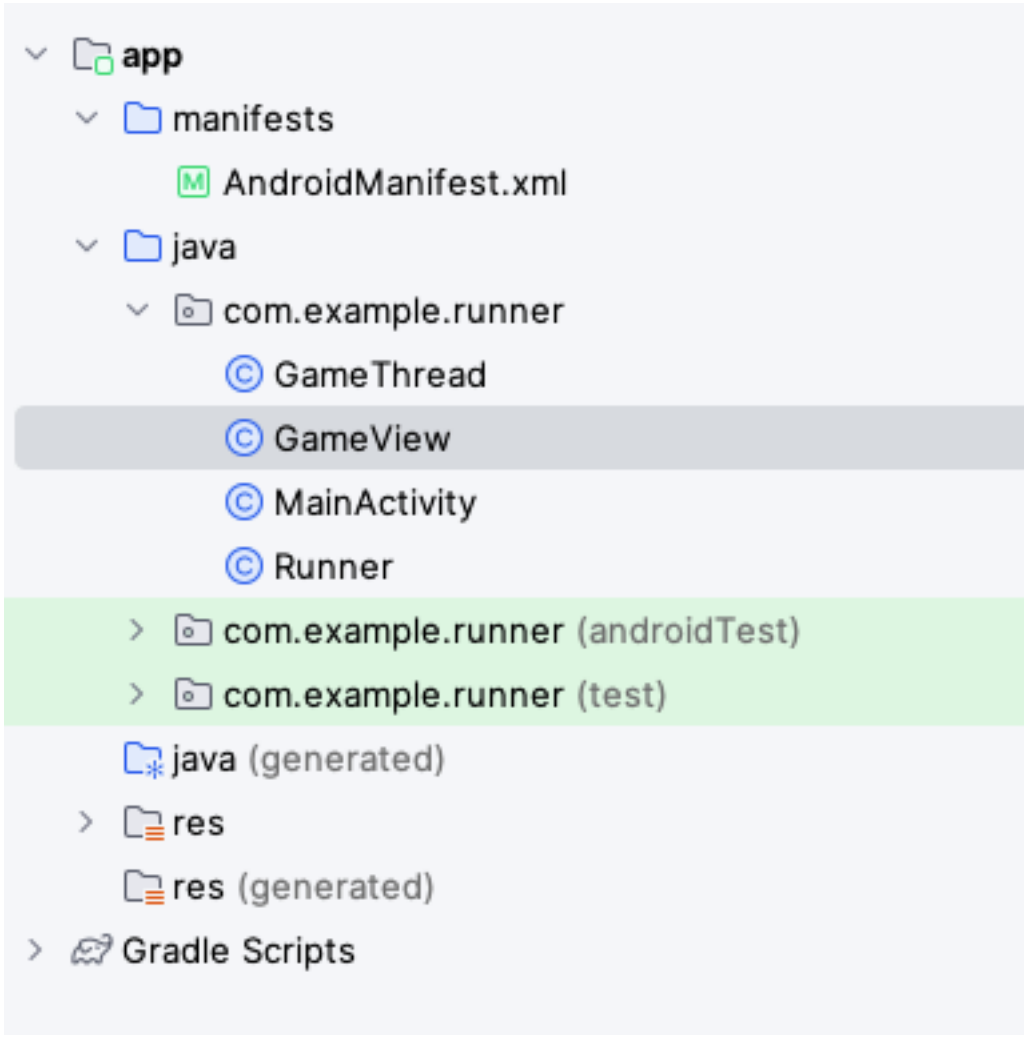
Klawiatura

# Animacja obiektu



Main

```
package com.example.runner;

import android.app.Activity;
import android.os.Bundle;

public class MainActivity extends Activity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      GameView gameView = new GameView(this);
      setContentView(gameView);
   }
}
```



# Animacja obiektu

```
package com.example.runner;

import android.graphics.Canvas;
import android.view.SurfaceHolder;

public class GameThread extends Thread {
    private SurfaceHolder surfaceHolder;
    private GameView gameView;
    private boolean running;

    public GameThread(SurfaceHolder holder, GameView view) {
        surfaceHolder = holder;
        gameView = view;
    }

    public void setRunning(boolean run) {
        running = run;
    }

    @Override
    public void run() {
        Canvas canvas;
        while (running) {
            canvas = null;
            try {
                canvas = surfaceHolder.lockCanvas();
                synchronized (surfaceHolder) {
                    gameView.update();
                    gameView.draw(canvas);
                }
            } finally {
                if (canvas != null)
                    surfaceHolder.unlockCanvasAndPost(canvas);
            }

            try {
                sleep(16); // ~60 FPS
            } catch (InterruptedException e) {}
        }
    }
}
```

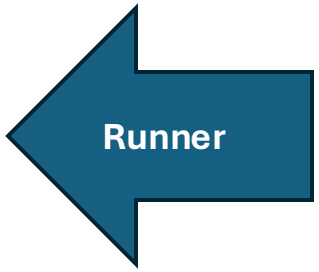
**GameThread**

**GameView**

```
package com.example.runner;

import android.content.Context;
import android.graphics.Canvas;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.MotionEvent;

public class GameView extends SurfaceView implements SurfaceHolder.Callback {
    private GameThread thread;
    private Runner runner;

    public GameView(Context context) {
        super(context);
        getHolder().addCallback(this);
        thread = new GameThread(getHolder(), this);
        runner = new Runner();
        setFocusable(true);
    }

    @Override
    public void surfaceCreated(SurfaceHolder holder) {
        runner.setScreenSize(getWidth(), getHeight());
        thread.setRunning(true);
        thread.start();
    }

    public void update() {
        runner.update();
    }

    @Override
    public void draw(Canvas canvas) {
        super.draw(canvas);
        if (canvas != null) {
            canvas.drawRGB(255, 255, 255); // Białe tło
            runner.draw(canvas);
        }
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        if (event.getAction() == MotionEvent.ACTION_DOWN) {
            runner.resetPosition();
        }
        return true;
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        boolean retry = true;
        thread.setRunning(false);
        while (retry) {
            try {
                thread.join();
                retry = false;
            } catch (InterruptedException e) {}
        }
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {}
}
```

# Animacja obiektu

```
package com.example.runner;

import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;

public class Runner {
    private int x, y, size;
    private int dx = 8; // prędkość w poziomie
    private int dy = 10; // prędkość w pionie
    private Paint paint;
    private int screenWidth = 1080;
    private int screenHeight = 1920;

    public Runner() {
        x = 100;
        y = 100;
        size = 100;

        paint = new Paint();
        paint.setColor(0xFFFF5722); // pomarańczowy
    }

    public void update() {
        x += dx;
        y += dy;

        // Odbicie od lewej/prawej
        if (x <= 0 || x + size >= screenWidth) {
            dx = -dx;
        }

        // Odbicie od góry/dół
        if (y <= 0 || y + size >= screenHeight) {
            dy = -dy;
        }
    }

    public void resetPosition() {
        x = 100;
        y = 100;
    }

    public void draw(Canvas canvas) {
        canvas.drawRect(new Rect(x, y, x + size, y + size), paint);
    }

    // Te metody wywołaj w GameView po poznaniu wymiarów ekranu:
    public void setScreenSize(int width, int height) {
        this.screenWidth = width;
        this.screenHeight = height;
    }
}
```

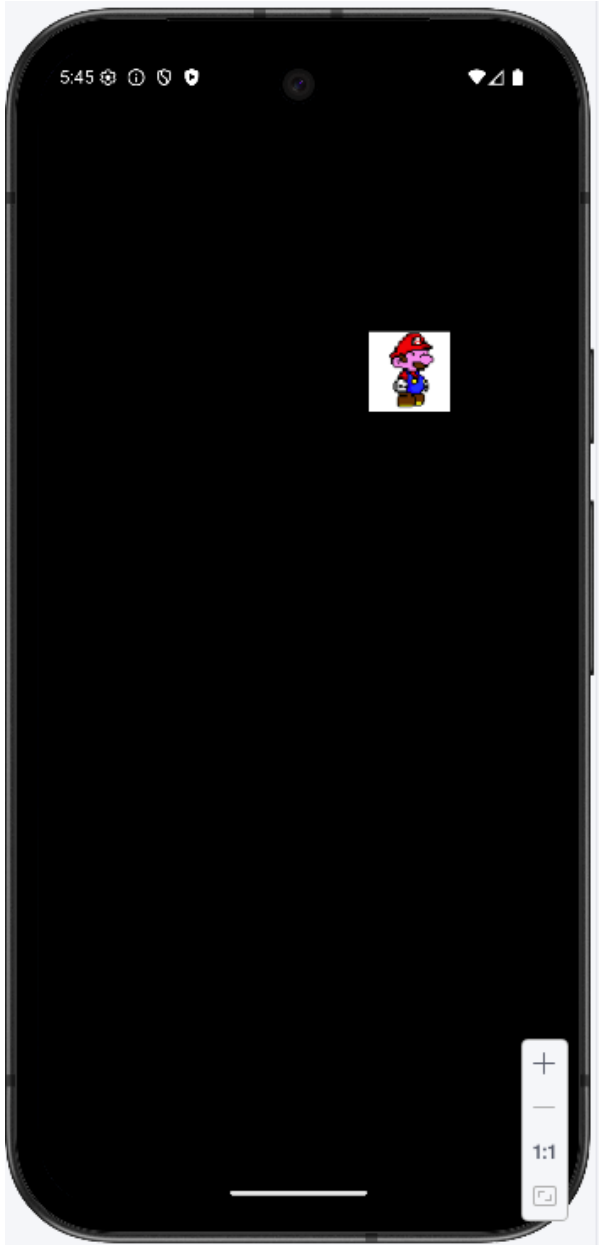



**Efekt?**

**Kwadrat odbija się od krawędzi okna. Porusza się w kierunku wyznaczonym przez odbicie. Można to wykorzystać w grach typu Pinball, Arkanoid czy w zmodyfikowanej wersji Tetrisa.**

# Dodawanie dźwięków do gry

```
app/
 ├─ java/
 │  └─ com.example.soundgame/
 │     ├─ MainActivity.java
 │     ├─ GameView.java
 │     └─ Player.java
 ├─ res/
 │  ├─ drawable/
 │  │  └─ player.png
 │  ├─ raw/
 │  │  ├─ shot.mp3
 │  │  └─ music.mp3
```



# Dodawanie dźwięków do gry

Prezentowany przykład zawiera muzykę w tle. Za każdym ruchem postaci słychać strzał.

```
package com.example.soundgame;

import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Player {
    private Bitmap bitmap;
    private int x = 100;
    private int y = 600;

    public Player(Bitmap bitmap) {
        this.bitmap = bitmap;
    }

    public void setX(int x) {
        this.x = x;
    }

    public void draw(Canvas canvas) {
        canvas.drawBitmap(bitmap, x - bitmap.getWidth() / 2, y, null);
    }
}
```

Player

MainActivity

```
package com.example.soundgame;

import android.app.Activity;
import android.media.MediaPlayer;
import android.os.Bundle;

public class MainActivity extends Activity {
    private MediaPlayer bgMusic;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        GameView gameView = new GameView(this);
        setContentView(gameView);

        bgMusic = MediaPlayer.create(this, R.raw.music);
        bgMusic.setLooping(true);
        bgMusic.start();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (bgMusic != null) {
            bgMusic.stop();
            bgMusic.release();
        }
    }
}
```

# Dodawanie dźwięków do gry

```
package com.example.soundgame;

import android.app.Activity;
import android.media.MediaPlayer;
import android.os.Bundle;

public class MainActivity extends Activity {
    private MediaPlayer bgMusic;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        GameView gameView = new GameView(this);
        setContentView(gameView);

        bgMusic = MediaPlayer.create(this, R.raw.music);
        bgMusic.setLooping(true);
        bgMusic.start();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (bgMusic != null) {
            bgMusic.stop();
            bgMusic.release();
        }
    }
}
```

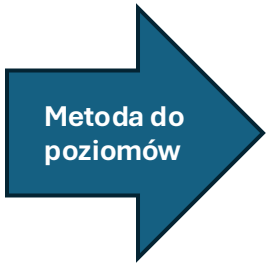
GameThread

GameView

```
package com.example.soundgame;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.media.SoundPool;
import android.media.AudioManager;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

public class GameView extends SurfaceView implements
SurfaceHolder.Callback {
    private GameThread thread;
    private Player player;
    private SoundPool soundPool;
    private int shotSound;

    public GameView(Context context) {
        super(context);
        getHolder().addCallback(this);
        thread = new GameThread(getHolder(), this);

        Bitmap playerBitmap =
BitmapFactory.decodeResource(getResources(),
R.drawable.player);
        player = new Player(playerBitmap);

        soundPool = new SoundPool(5, AudioManager.STREAM_MUSIC,
0);
        shotSound = soundPool.load(context, R.raw.shot, 1);

        setFocusable(true);
    }

    @Override
    public void surfaceCreated(SurfaceHolder holder) {
        thread.setRunning(true);
        thread.start();
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        player.setX((int) event.getX());
        soundPool.play(shotSound, 1, 1, 0, 0, 1);
        return true;
    }

    public void update() {
        // Można dodać logikę gry
    }

    @Override
    public void draw(Canvas canvas) {
        super.draw(canvas);
        if (canvas != null) {
            canvas.drawRGB(0, 0, 0);
            player.draw(canvas);
        }
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        boolean retry = true;
        thread.setRunning(false);
        while (retry) {
            try {
                thread.join();
                retry = false;
            } catch (InterruptedException e) { }
        }
        soundPool.release();
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int
width, int height) {}
}
```

# Poziomy gry i kolejne plansze

```
public class Level {
    int levelNumber;
    String mapData; // np. układ planszy jako tekst, JSON, itp.
    int difficulty;

    public Level(int levelNumber, String mapData, int difficulty) {
        this.levelNumber = levelNumber;
        this.mapData = mapData;
        this.difficulty = difficulty;
    }

    // Gettery i settery
}
```


```
List<Level> levels = new ArrayList<>();

private void loadLevels() {
    levels.add(new Level(1, "mapa1.json", 1));
    levels.add(new Level(2, "mapa2.json", 2));
    levels.add(new Level(3, "mapa3.json", 3));
}
```

Pliki z rozmieszczeniem elementów znajdują się w plikach JSON w folderze Assets lub w Res. Pliki JSON zawierają mapę, tzn. rozmieszczenie elementów na widoku.

```
{
  "width": 10,
  "height": 10,
  "tiles": [
    [1, 0, 0, 1, ...],
    ...
  ]
}
```

```
private String loadMapFromAssets(String fileName) {
    try {
        InputStream is = getAssets().open(fileName);
        int size = is.available();
        byte[] buffer = new byte[size];
        is.read(buffer);
        is.close();
        return new String(buffer, "UTF-8");
    } catch (IOException ex) {
        ex.printStackTrace();
        return null;
    }
}
```

# Gettery i Settery

```
public class Level {
    private int levelNumber;

    // Konstruktor
    public Level(int levelNumber) {
        this.levelNumber = levelNumber;
    }

    // Getter
    public int getLevelNumber() {
        return levelNumber;
    }

    // Setter
    public void setLevelNumber(int levelNumber) {
        this.levelNumber = levelNumber;
    }
}
```

**Getter** (getLevelNumber) — zwraca wartość prywatnego pola levelNumber.
**Setter** (setLevelNumber) — ustawia nową wartość pola levelNumber.

```
Level level = new Level(1);

// odczyt poziomu
int current =
level.getLevelNumber();  // wynik: 1

// zmiana poziomu
level.setLevelNumber(2);

// ponowny odczyt
int updated =
level.getLevelNumber();  // wynik: 2
```

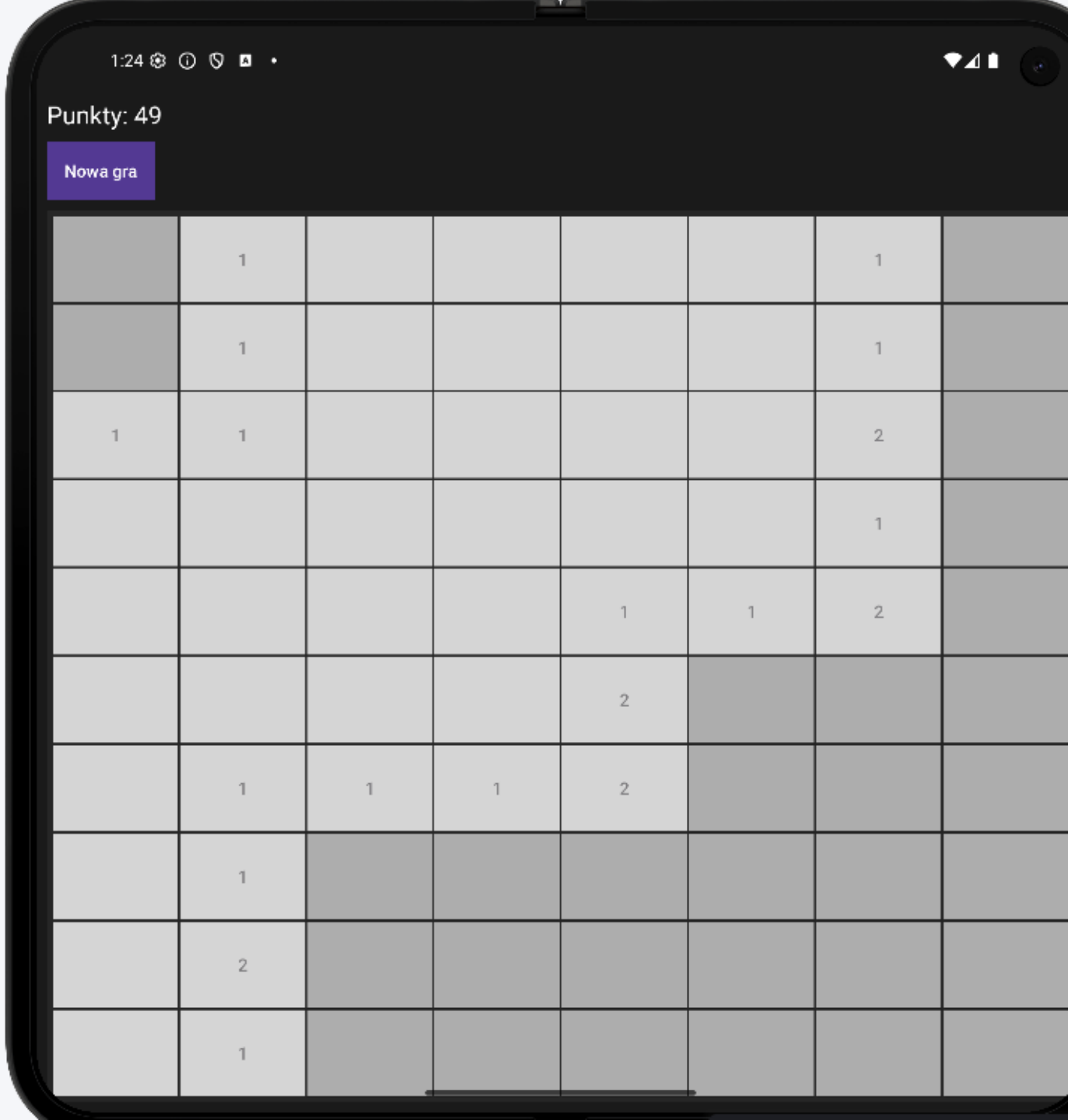
# Podsumowanie - pytania

- Co należy dodać w manifeście aby mieć dostęp do plików w telefonie?
- Do czego służy PICK_VIDEO_REQUEST?
- Jak zwykle wygląda struktura plików w grze?
- Czy można dopisać ruch postaci za pomocą klawiszy czy tylko przez dotyk?
- Czy do projektowania gry potrzebny jest Canvas?
- W jakim pliku przechowywana jest mapa gry?
- Do czego służą gettery i settery?

# Gry do samodzielnego wykonania

## Zadanie 1

Prosta gra „Saper”, w której użytkownik odkrywa obszar. Jeżeli trafi na „bombę” to koniec gry. Jeżeli na wybrane pola to będzie miał przyznane punkty.

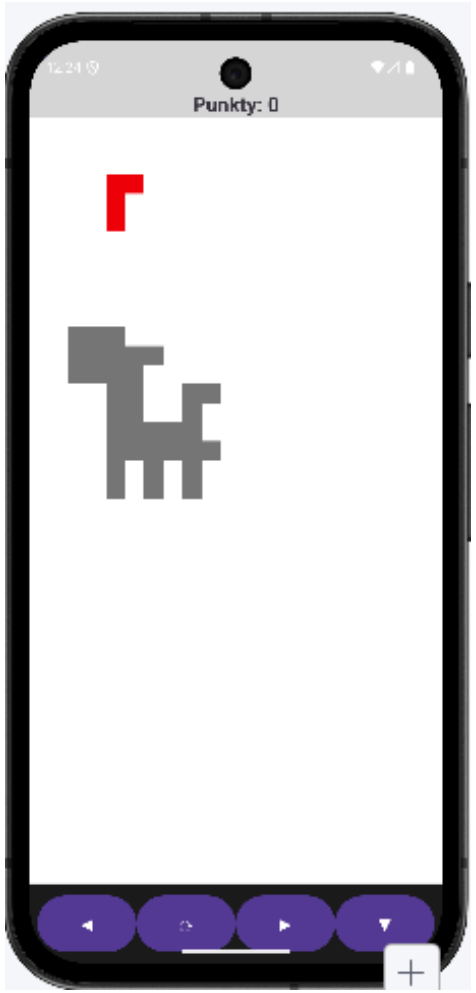


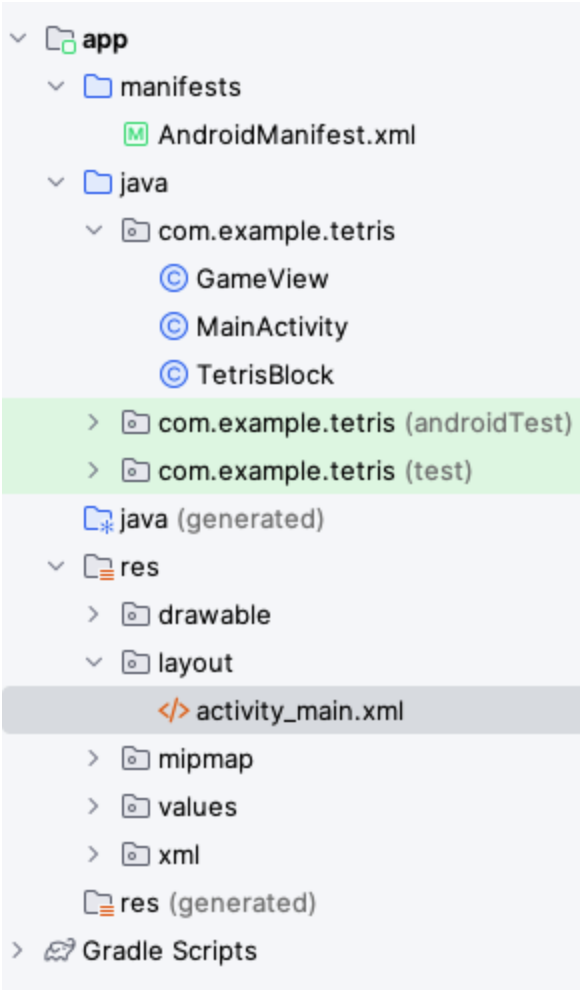
# Gry do samodzielnego wykonania

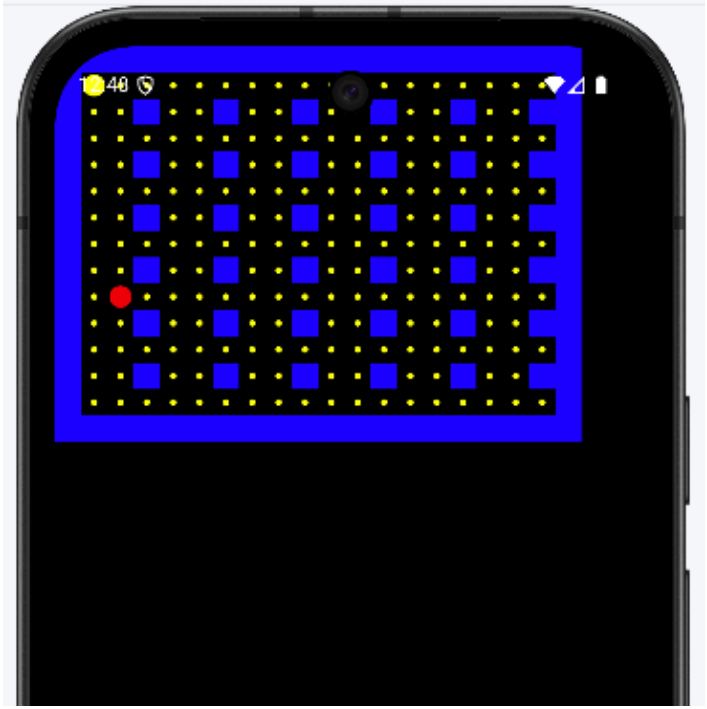
## Zadanie 2

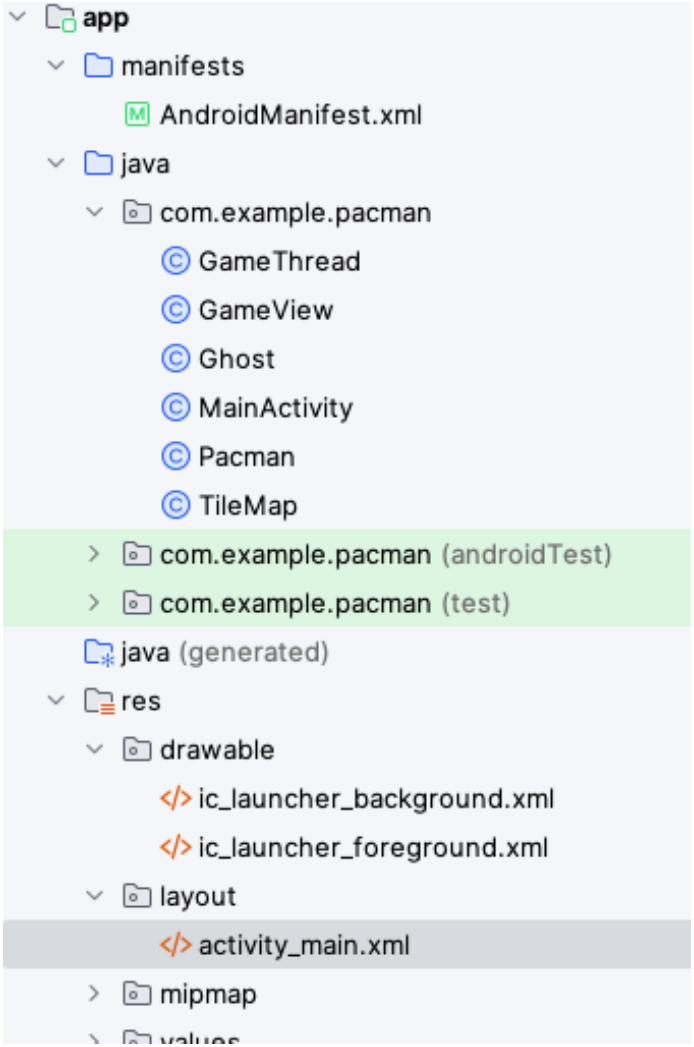
Gra Tetris, w której użytkownik steruje spadającymi klockami za pomocą klawiatury (BottomNavigation).

Do projektu możesz dodać różne kształty i kolory klocków, zegar oraz punkty za każdą wypełnioną linię.








# Gry do samodzielnego wykonania

**Zadanie 3 (zadanie na 6)**

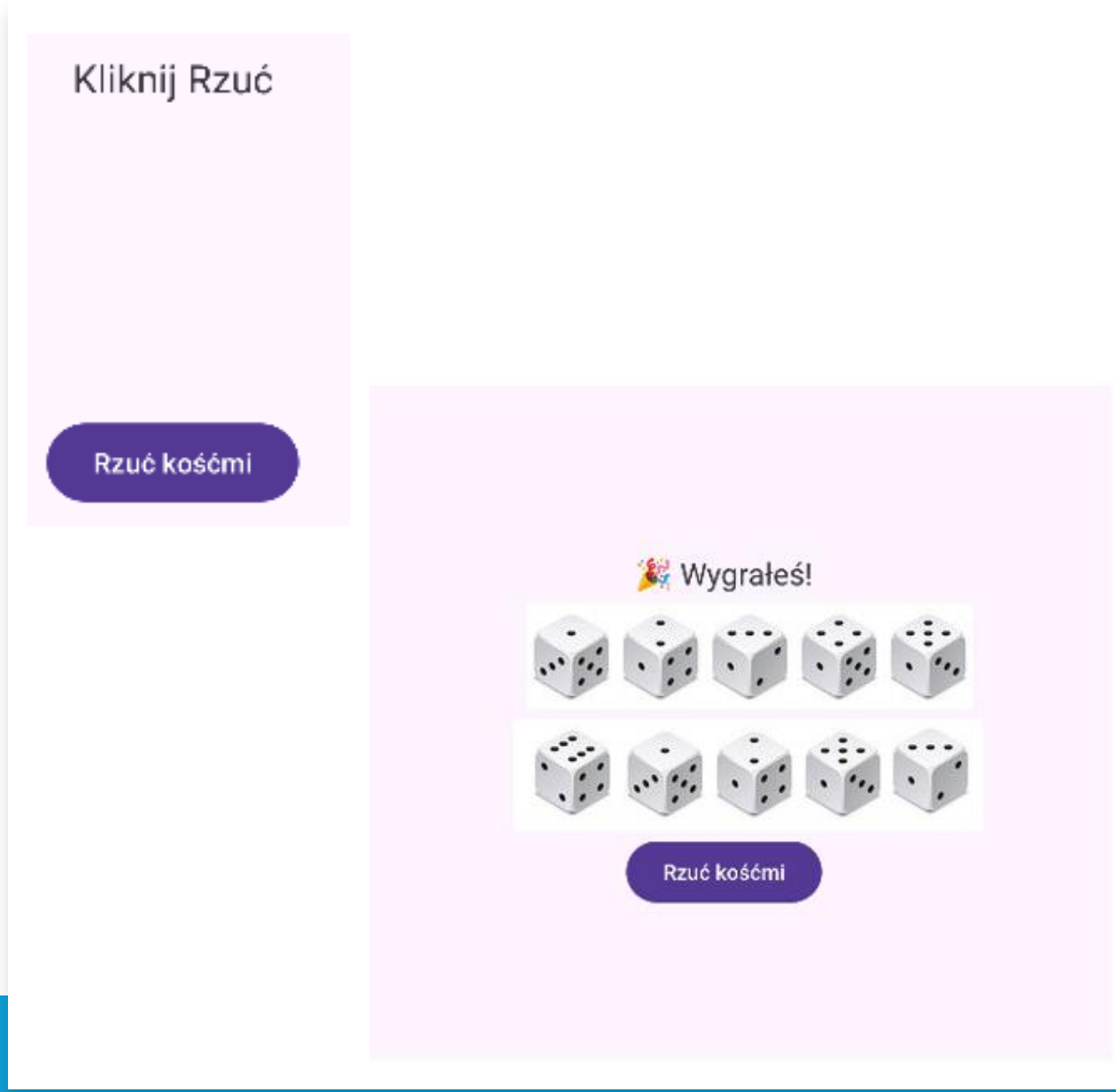
Gra „Pac-Man”, w której sterujemy „Buźką”. Zjada kropki i musi wyczyścić cały labirynt. Do projektu dodaj „Duszki”, które chodzą po labiryncie. Za znaczne rozwinięcie projektu ocean 6.

# Gry do samodzielnego wykonania

**Zadanie 4**

Gra w „Kości”. Grasz przeciwko komputerowi. Do projektu potrzebujesz zdjęć 6 ścian kostki. Rozwiń grę za pomocą zliczania wyrzuconych „oczek”.

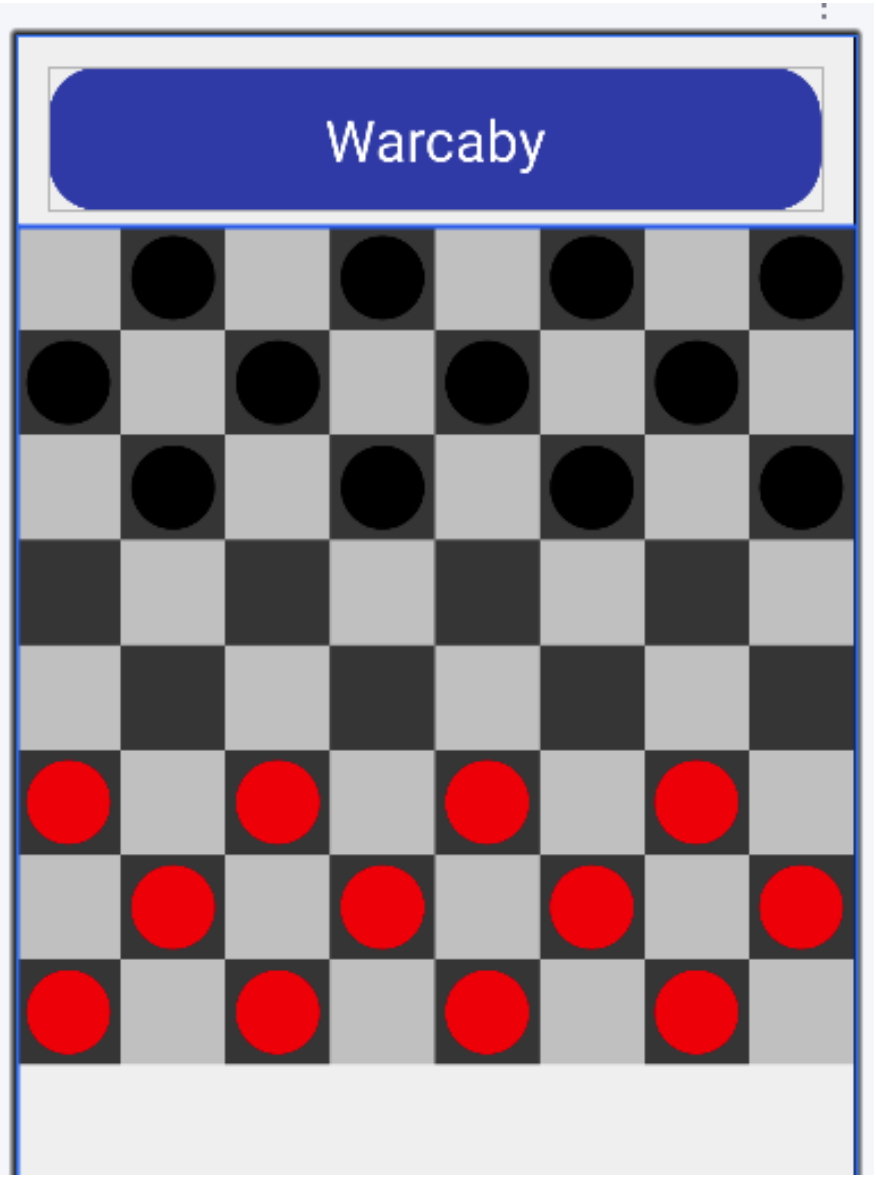
Gra zawiera dwie Activity.



# Gry do samodzielnego wykonania

**Zadanie 5**

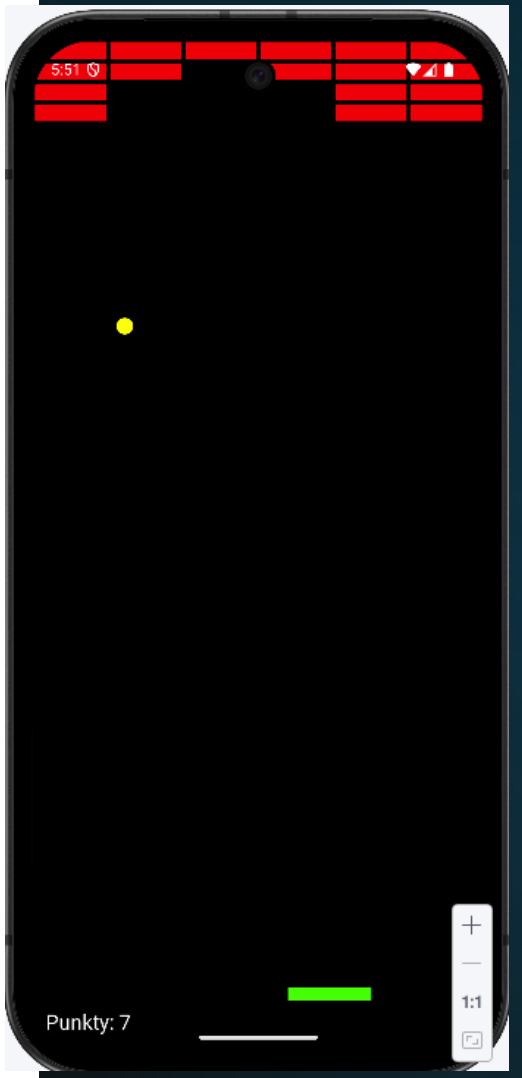
Gra w warcaby. Gra dwóch fizycznych użytkowników. Użytkownik zaznacza swoje „kółko” a następnie wskazuje miejsce docelowe. Po zajęciu pola przeciwnika jego „kółko” znika. Do projektu dodaj licznik zliczający punkty.



# Gry do samodzielnego wykonania

**Zadanie 6 (zadanie na 6)**

Na podstawie załączonego widoku wykonaj grę „Pinball” z dołączonymi przeszkodami. Za każdą zbitą cegłę użytkownik otrzymuje 1 punkt. Logika gry jest rozpisana na 3 klasy.



# Gry do samodzielnego wykonania

**Zadanie 7 (na 6)**

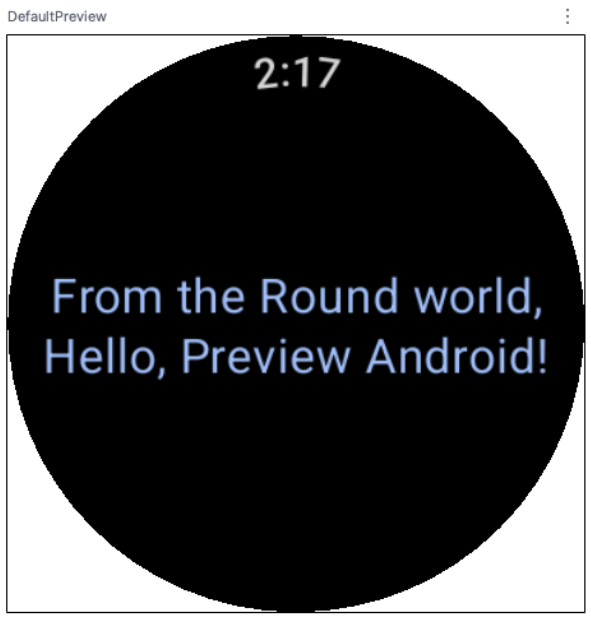
Klasyczna gra „Koło Fortuny”. Użytkownik losuje nagrodę a następnie odgaduje literę z hasła. Gdy udzieli poprawnej odpowiedzi wyświetla się komunikat.

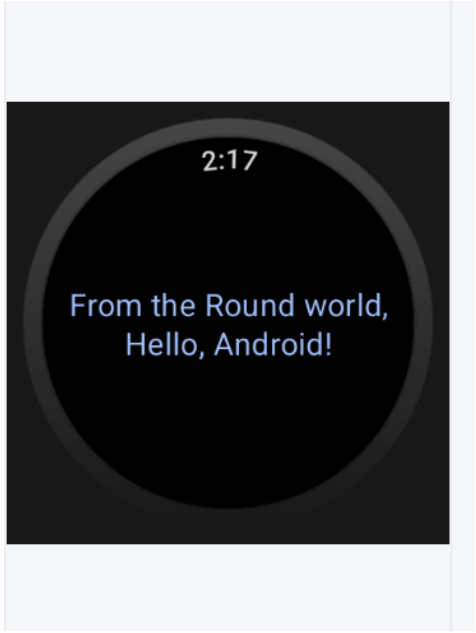
# Wykorzystanie szablonów dla WearOS oraz AndroidAuto

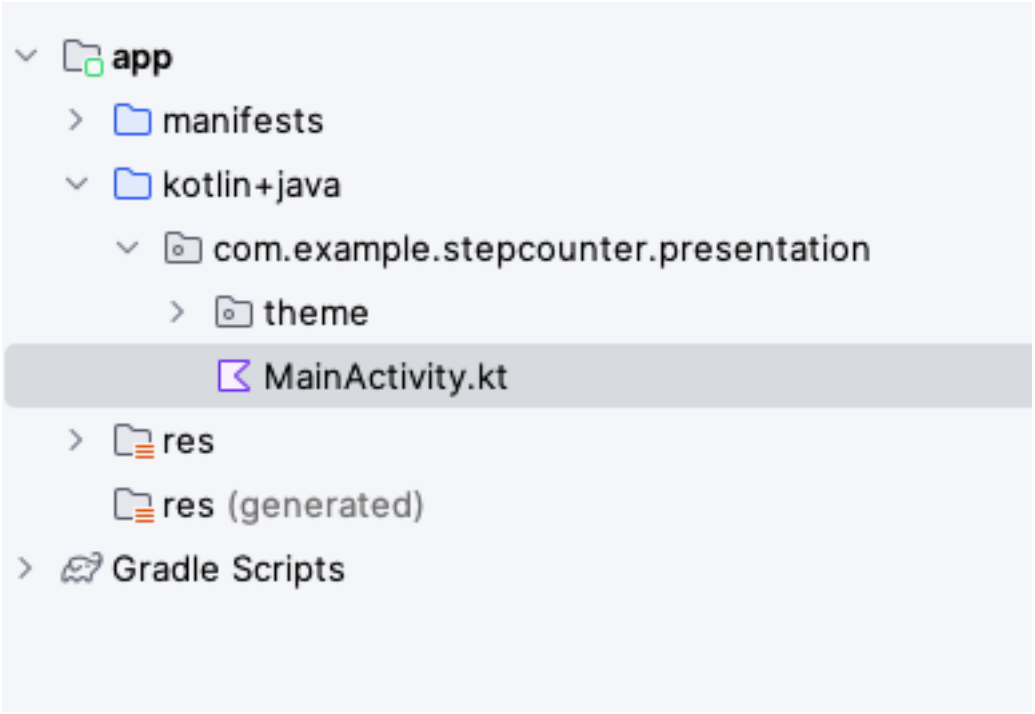
W najnowszych wersjach Android Studio szablony dla WearOS są dostępne jedynie dla języka Kotlin.

```
package com.example.stepcounter.presentation

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import
androidx.core.splashscreen.SplashScreen.Companion.insta
llSplashScreen
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Preview
import androidx.wear.compose.material.MaterialTheme
import androidx.wear.compose.material.Text
import androidx.wear.compose.material.TimeText
import androidx.wear.tooling.preview.devices.WearDevices
import com.example.stepcounter.R
import
com.example.stepcounter.presentation.theme.Stepcounter
Theme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        installSplashScreen()

        super.onCreate(savedInstanceState)

        setTheme(android.R.style.Theme_DeviceDefault)

        setContent {
            WearApp("Android")
        }
    }
}

@Composable
fun WearApp(greetingName: String) {
    StepcounterTheme {
        Box(
            modifier = Modifier
                .fillMaxSize()
                .background(MaterialTheme.colors.background),
            contentAlignment = Alignment.Center
        ) {
            TimeText()
            Greeting(greetingName = greetingName)
        }
    }
}

@Composable
fun Greeting(greetingName: String) {
    Text(
        modifier = Modifier.fillMaxWidth(),
        textAlign = TextAlign.Center,
        color = MaterialTheme.colors.primary,
        text = stringResource(R.string.hello_world,
greetingName)
    )
}

@Preview(device = WearDevices.SMALL_ROUND,
showSystemUi = true)
@Composable
fun DefaultPreview() {
    WearApp("Preview Android")
}
```

# Wykorzystanie szablonów dla WearOS oraz AndroidAuto



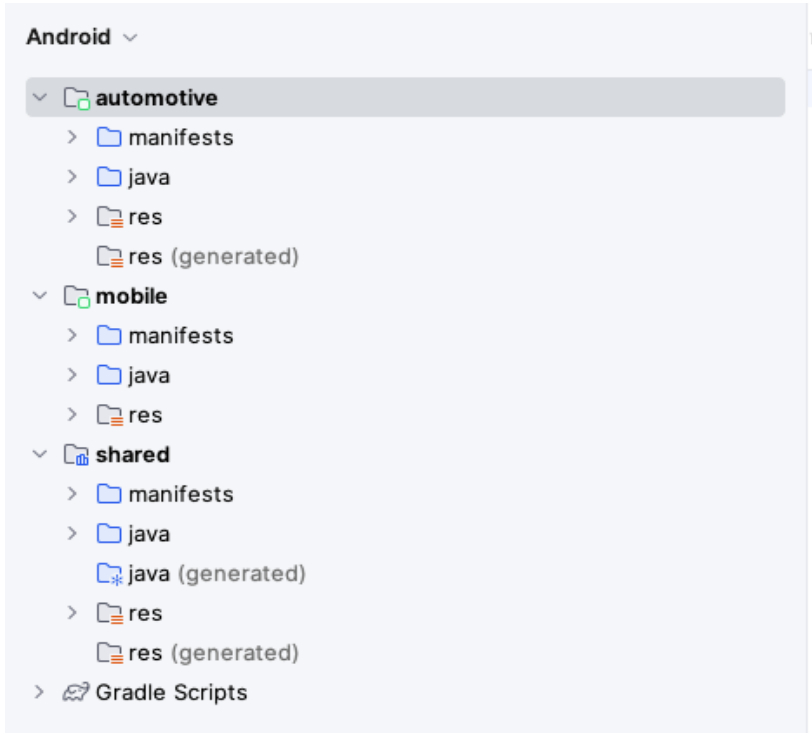



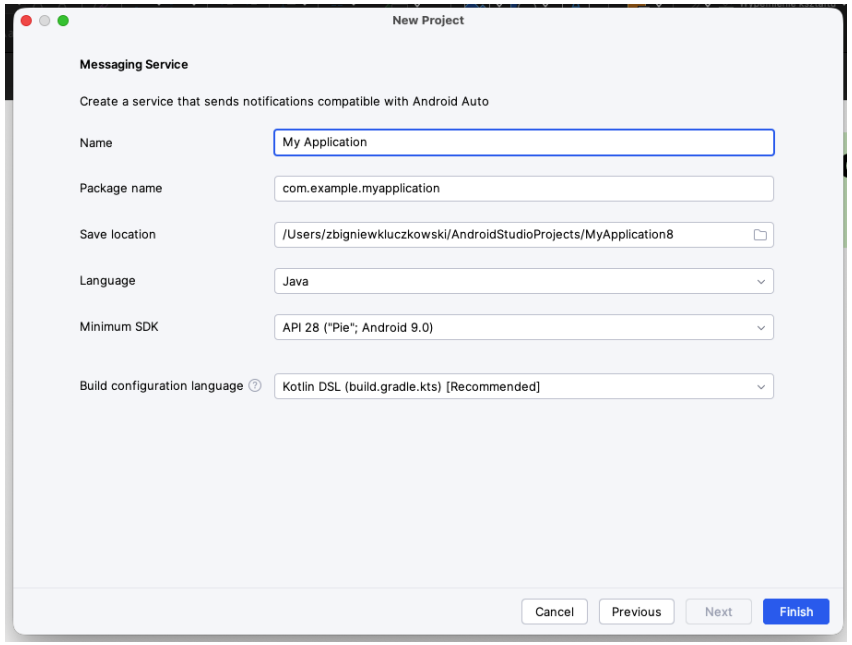



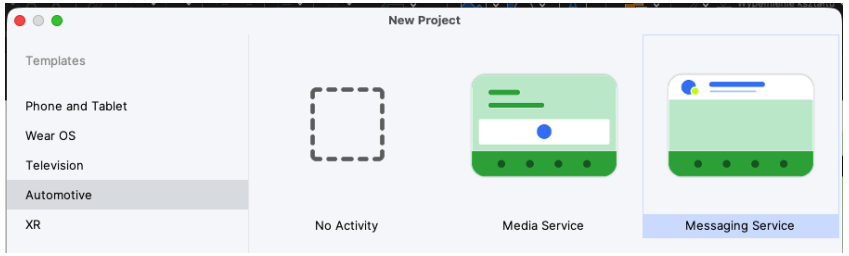
# Wykorzystanie szablonów dla WearOS oraz AndroidAuto

W przypadku Android Auto jest możliwość skorzystania z Javy (jeszcze).

Projekty z wykorzystaniem Kotlina znajdziesz w drugiej prezentacji.



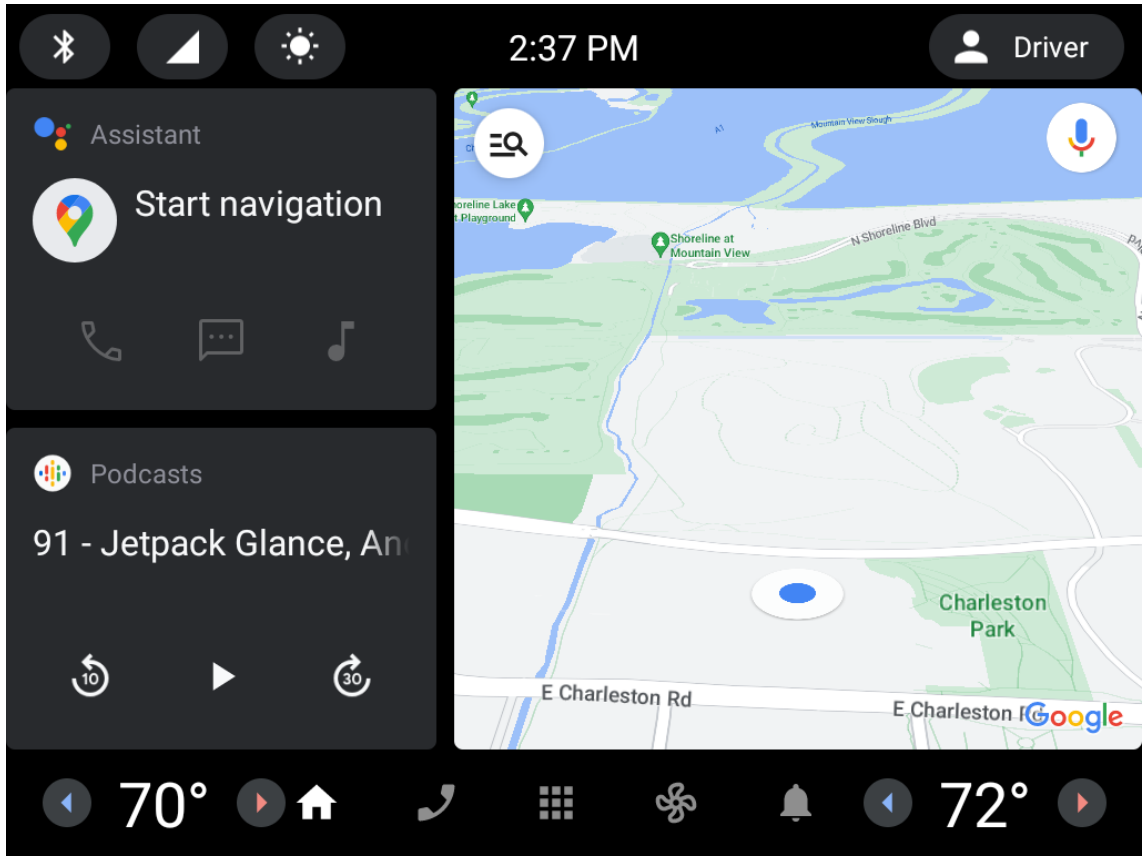
# Wykorzystanie szablonów dla WearOS oraz AndroidAuto

**Do napisania aplikacji służącej do nawigacji potrzebujesz:**

"androidx.car.app:app:1.4.0" -

Umożliwia tworzenie interfejsu użytkownika zgodnego z Android Auto, np.:

- Nawigacja (NavigationTemplate)
- Lista miejsc (PlaceListMapTemplate)
- Przycisk „Start” / „Zakończ trasę”
- Integracja z mapami i głosem



# Wykorzystanie szablonów dla WearOS oraz AndroidAuto

implementation 'com.google.android.gms:play-services-maps:18.2.0’

Umożliwia:

- Wyświetlanie mapy
- Tworzenie tras
- Interakcję z lokalizacją

implementation 'com.google.android.gms:play-services-location:21.0.1'

Obsługuje:

- Pobieranie bieżącej lokalizacji GPS
- Aktualizacje lokalizacji w tle
- Obsługę geofencing (obszary)

implementation 'com.mapbox.navigation:android:2.16.1'

- Pełna nawigacja zakręt-po-zakręcie
- Własne style mapy
- Możliwość działania offline

TTS (Text-To-speech) – wbudowana w Androida (nie trzeba dodatkowej biblioteki)

Retrofit lub OkHTTP – do pobierania tras API:

implementation 'com.squareup.retrofit2:retrofit:2.9.0'

implementation 'com.squareup.okhttp3:okhttp:4.12.0’

GSON lub Moshi – do przetwarzania danych z JSON

implementation 'com.google.code.gson:gson:2.10.1’

LifeCycle i ViewModel do zarządzania logiką:

implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.7.0"

implementation "androidx.lifecycle:lifecycle-livedata-ktx:2.7.0"

# Java i Android na egzaminie INF.04

Na początek dobre rady:

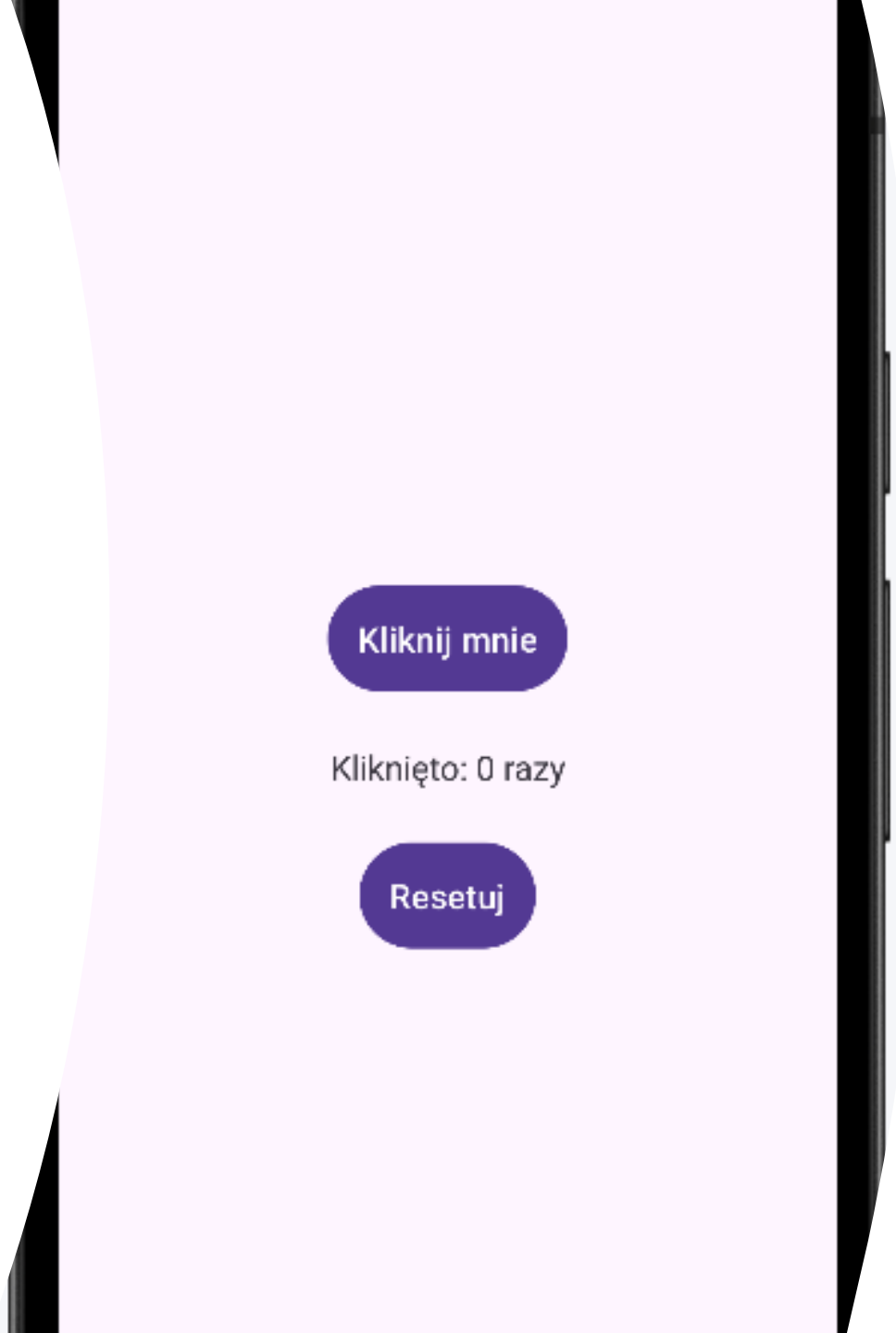
- Przed egzaminem poznaj strukturę projektu i składnię Javy.
- Zwracaj uwagę na poprawne nazwy ID i layoutów.
- Używaj emulatora, aby szybko testować aplikację.
- Pamiętaj o obsłudze zdarzeń (np. kliknięcia przycisków).
- Stosuj czytelne i prostolinijne rozwiązania.

**Na egzaminie INF.04 ważne jest opanowanie podstaw — tworzenie layoutu i obsługa interakcji.**

# Proponowane zadania

| Typ aplikacji | Elementy wymagane |
| --- | --- |
| Kalkulator | Obsługa działań, wyświetlanie wyniku, walidacja danych |
| Formularz rejestracyjny | Pola tekstowe, Spinner, CheckBox, zapis danych |
| Lista zadań (To-Do List) | RecyclerView, dodawanie/usuwanie, zapis stanu |
| Quiz z punktacją | Pytania, CheckBox/RadioButton, wynik, ew. zapis do pliku |
| Generator liczby losowej | Random, zakres, przycisk, wyświetlenie liczby |
| Aplikacja z bazą SQLite | Dodawanie/wyświetlanie/usuwanie danych |
| Aplikacja pogodowa (na API) | Fetch JSON, wyświetlanie danych |

# Typowe zadanie?




**Temat:** Stwórz prostą aplikację mobilną „Licznik kliknięć”.

**Opis zadania:**

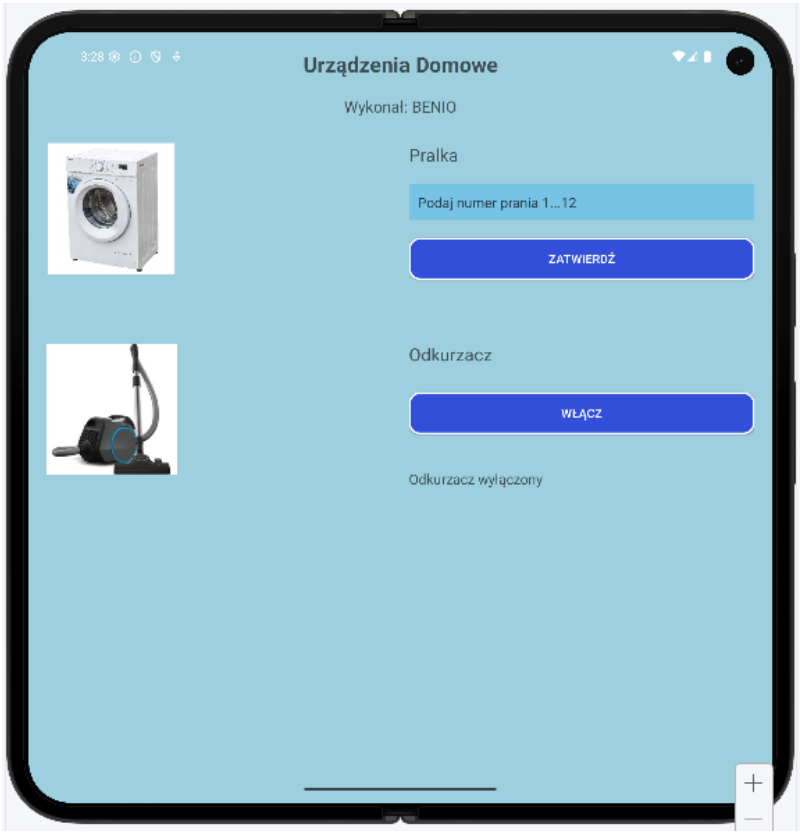
1. **Interfejs:**
    1. Na ekranie znajduje się przycisk **„Kliknij mnie”**.
    2. Pod przyciskiem jest wyświetlany tekst pokazujący, ile razy przycisk został kliknięty, np. „Kliknięto: 0 razy”.
2. **Funkcjonalność:**
    1. Po każdym kliknięciu przycisku licznik zwiększa się o 1 i aktualizuje tekst na ekranie.
    2. Aplikacja powinna działać bez błędów.
3. **Wymagania techniczne:**
    1. Layout utwórz w pliku XML (np. activity_main.xml).
    2. Logikę napisz w pliku Java (MainActivity.java).
    3. Użyj przycisku (Button) i pola tekstowego (TextView).
    4. Wartość licznika powinna być zapamiętana podczas działania aplikacji.

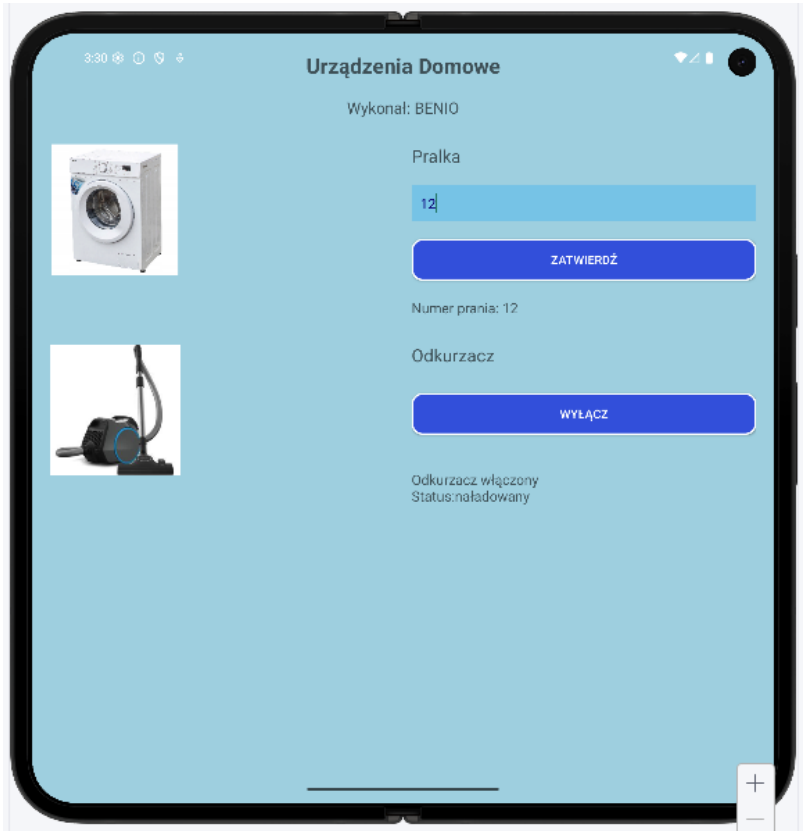
# Kod zadania

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

xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:padding="24dp"
   android:gravity="center"
   android:layout_width="match_parent"
   android:layout_height="match_parent">


   <Button
      android:id="@+id/buttonClick"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Kliknij mnie"
      android:padding="16dp"
      android:textSize="18sp"/>


   <TextView
      android:id="@+id/textViewCount"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Kliknięto: 0 razy"
      android:textSize="18sp"
      android:layout_marginTop="24dp"/>


   <Button
      android:id="@+id/buttonReset"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Resetuj"
      android:layout_marginTop="24dp"
      android:padding="16dp"
      android:textSize="18sp"/>


</LinearLayout>
```

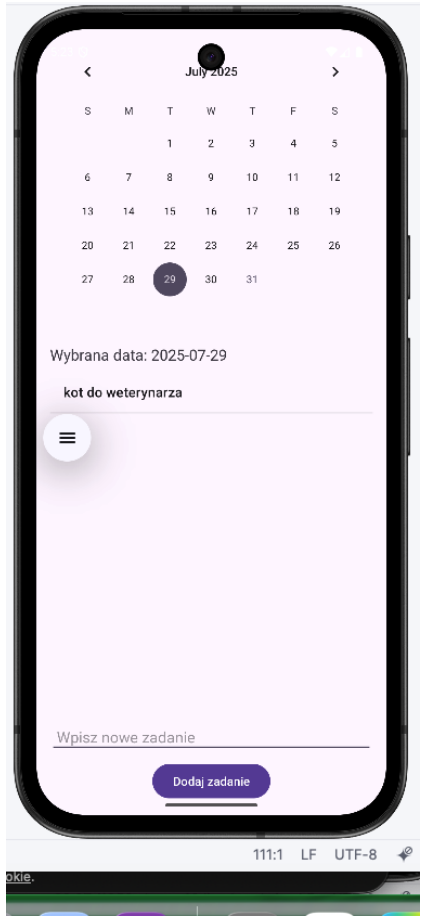
```
package com.example.licznikklikniec;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import
androidx.appcompat.app.AppCompatA
ctivity;

public class MainActivity extends
AppCompatActivity{

   private int count = 0;
   private TextView textViewCount;

   @Override
   protected void onCreate(Bundle
savedInstanceState){

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

      textViewCount =
findViewById(R.id.textViewCount);
      Button buttonClick =
findViewById(R.id.buttonClick);
      Button buttonReset =
findViewById(R.id.buttonReset);

      buttonClick.setOnClickListener(new
View.OnClickListener(){
         @Override
         public void onClick(View v){
            count++;
            updateCountText();
         }
      });


buttonReset.setOnClickListener(new
View.OnClickListener(){
         @Override
         public void onClick(View v){
            count = 0;
            updateCountText();
         }
      });

      updateCountText();
   }

   private void updateCountText(){
      textViewCount.setText("Kliknięto: " +
count + " razy");
   }
}
```

# „Urządzenia domowe”



# Kalendarz z zadaniami




Aplikacja w formie kalendarza i listy zadań. Po kliknięciu na datę odświeża się widok listy z zadaniami. Użytkownik może dodać własne.

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

xmlns:android="http://schemas.android.com/
apk/res/android"
  android:orientation="vertical"
  android:padding="16dp"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <CalendarView
    android:id="@+id/calendarView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

  <TextView
    android:id="@+id/selectedDateText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Wybrana data:"
    android:textSize="18sp"
    android:paddingTop="16dp"/>

  <ListView
    android:id="@+id/tasksListView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:paddingTop="8dp"
    android:dividerHeight="1dp"/>

  <EditText
    android:id="@+id/newTaskEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Wpisz nowe zadanie"
    android:inputType="text"
    android:padding="8dp"
    android:layout_marginTop="8dp"/>

  <Button
    android:id="@+id/addTaskButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Dodaj zadanie"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="8dp"/>

</LinearLayout>
```

# Kalendarz z zadaniami

```
package com.example.calendarapp;

import android.os.Bundle;
import android.text.TextUtils;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.CalendarView;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;

public class MainActivity extends AppCompatActivity{

  private CalendarView calendarView;
  private TextView selectedDateText;
  private ListView tasksListView;
  private EditText newTaskEditText;
  private Button addTaskButton;

  private HashMap<String, ArrayList<String>> tasksMap;
  private String currentSelectedDate;

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

    calendarView = findViewById(R.id.calendarView);
    selectedDateText = findViewById(R.id.selectedDateText);
    tasksListView = findViewById(R.id.tasksListView);
    newTaskEditText = findViewById(R.id.newTaskEditText);
    addTaskButton = findViewById(R.id.addTaskButton);

    prepareTasks();

    // Początkowa data: dzisiaj
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    currentSelectedDate = sdf.format(new Date());
    selectedDateText.setText("Wybrana data: " + currentSelectedDate);
    updateTasksForDate(currentSelectedDate);

    calendarView.setOnDateChangeListener((view, year, month, dayOfMonth) -> {
      currentSelectedDate = String.format("%04d-%02d-%02d", year, month + 1,
dayOfMonth);
      selectedDateText.setText("Wybrana data: " + currentSelectedDate);
      updateTasksForDate(currentSelectedDate);
    });

    addTaskButton.setOnClickListener(v -> {
      String newTask = newTaskEditText.getText().toString().trim();
      if (TextUtils.isEmpty(newTask)) {
        Toast.makeText(MainActivity.this, "Wpisz zadanie!",
Toast.LENGTH_SHORT).show();
        return;
      }

      // Dodaj zadanie do mapy
      ArrayList<String> tasks = tasksMap.get(currentSelectedDate);
      if (tasks == null) {
        tasks = new ArrayList<>();
        tasksMap.put(currentSelectedDate, tasks);
      }
      tasks.add(newTask);

      // Odśwież listę
      updateTasksForDate(currentSelectedDate);

      // Wyczyść pole tekstowe
      newTaskEditText.setText("");
      Toast.makeText(MainActivity.this, "Zadanie dodane",
Toast.LENGTH_SHORT).show();
    });
  }

  private void prepareTasks(){
    tasksMap = new HashMap<>();

    ArrayList<String> tasks1 = new ArrayList<>();
    tasks1.add("Spotkanie z klientem");
    tasks1.add("Wysłanie raportu");
    tasksMap.put("2025-07-31", tasks1);

    ArrayList<String> tasks2 = new ArrayList<>();
    tasks2.add("Przygotować prezentację");
    tasksMap.put("2025-08-01", tasks2);

    ArrayList<String> tasks3 = new ArrayList<>();
    tasks3.add("Dzień wolny");
    tasks3.add("Wyjazd na konferencję");
    tasksMap.put("2025-08-05", tasks3);
  }

  private void updateTasksForDate(String date) {
    ArrayList<String> tasks = tasksMap.get(date);

    if (tasks == null || tasks.isEmpty()) {
      tasks = new ArrayList<>();
      tasks.add("Brak zadań na ten dzień");
    }

    ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
        android.R.layout.simple_list_item_1, tasks);
    tasksListView.setAdapter(adapter);
  }
}
```

# Zapis do pliku lub bazy na egzaminie?

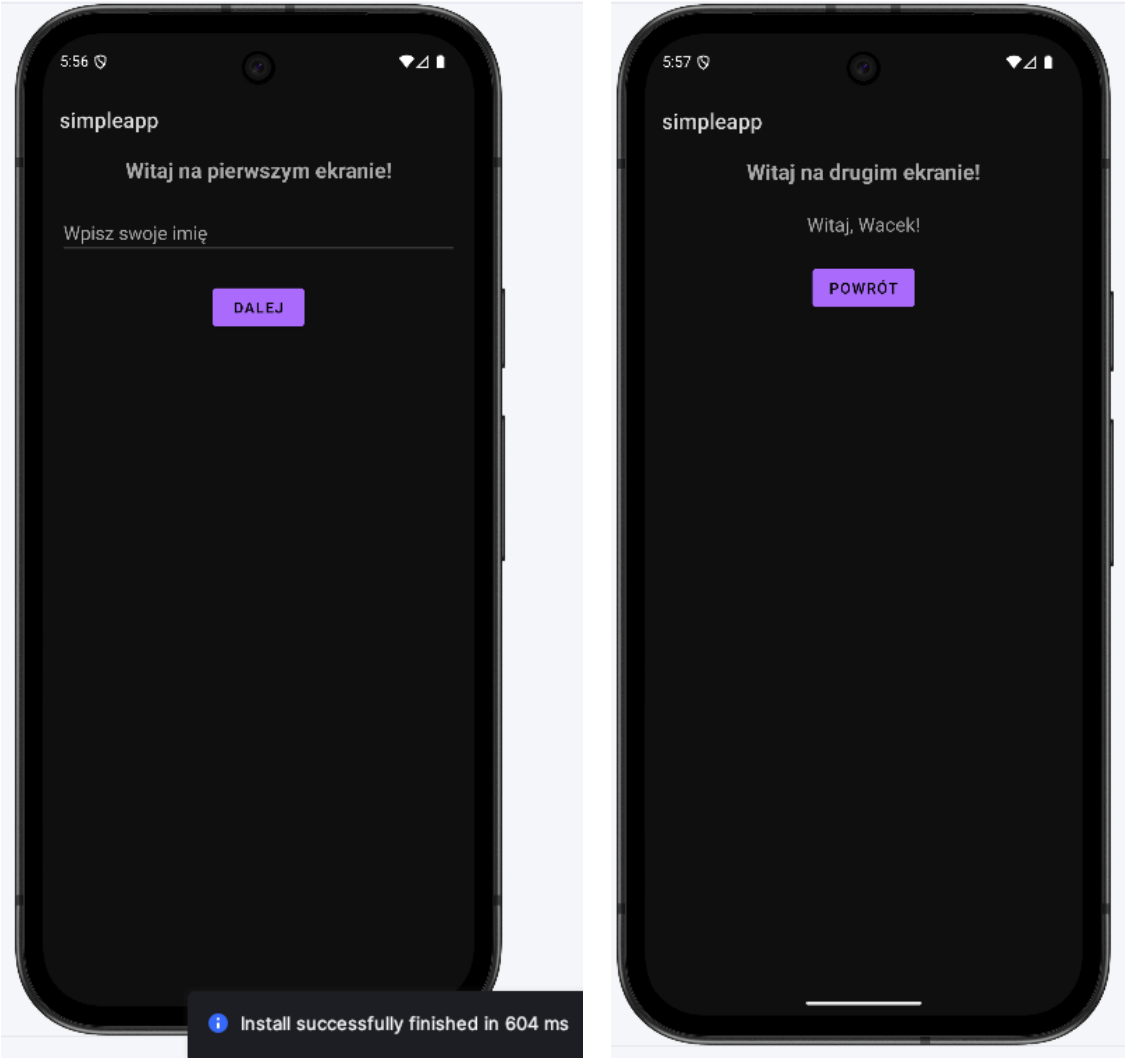
Tak, to jest możliwe, ale będzie to plik lub baza umieszczone w projekcie. W praktyce oznacza to projekt, który zawiera najwyżej kilka elementów w kilku plikach o długości kodu od kilkunastu do około 30 linii (najdłuższe będą pliki XML).

```
package com.example.calendarapp;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import
android.database.sqlite.SQLiteDatabase;
import
android.database.sqlite.SQLiteOpenHelper;

import java.util.ArrayList;

public class DBHelper extends
SQLiteOpenHelper {

  private static final String DATABASE_NAME =
"tasks.db";
  private static final int DATABASE_VERSION = 1;

  private static final String TABLE_TASKS =
"tasks";
  private static final String COLUMN_ID = "id";
  private static final String COLUMN_DATE =
"date";
  private static final String COLUMN_TASK =
"task";

  public DBHelper(Context context) {
    super(context, DATABASE_NAME, null,
DATABASE_VERSION);
  }

  @Override
  public void onCreate(SQLiteDatabase db) {
    String createTable = "CREATE TABLE " +
TABLE_TASKS + "(" +
        COLUMN_ID + " INTEGER PRIMARY KEY
AUTOINCREMENT, " +
        COLUMN_DATE + " TEXT, " +
        COLUMN_TASK + " TEXT)";
    db.execSQL(createTable);
  }

  @Override
  public void onUpgrade(SQLiteDatabase db, int
oldVersion, int newVersion) {
    // Jeśli zmiana wersji, usuń starą tabelę i
utwórz nową
    db.execSQL("DROP TABLE IF EXISTS " +
TABLE_TASKS);
    onCreate(db);
  }

  // Dodaj nowe zadanie do bazy
  public long addTask(String date, String task) {
    SQLiteDatabase db =
this.getWritableDatabase();
    ContentValues values = new
ContentValues();
    values.put(COLUMN_DATE, date);
    values.put(COLUMN_TASK, task);
    long id = db.insert(TABLE_TASKS, null,
values);
    db.close();
    return id;
  }

  // Pobierz listę zadań dla podanej daty
  public ArrayList<String> getTasks(String date) {}
    ArrayList<String> tasks = new ArrayList<>();
    SQLiteDatabase db =
this.getReadableDatabase();
    Cursor cursor = db.query(TABLE_TASKS,
        new String[]{COLUMN_TASK},
        COLUMN_DATE + "=?",
        new String[]{date},
        null, null, null);

    if (cursor != null) {
      while (cursor.moveToNext()) {
        String task =
cursor.getString(cursor.getColumnIndexOrThrow
(COLUMN_TASK));
        tasks.add(task);
      }
      cursor.close();
    }

    db.close();
    return tasks;
  }
```

# Zapis do pliku lub bazy na egzaminie?

Polecenie może zawierać wykonanie zrzutów ekranu. To oznacza, że nie praktykuje się zadań wykorzystujących skomplikowane sposoby zapisu.

„Skomplikowanie zadania” dotyczy zwykle układu (plików XML) lub nawigacji pomiędzy kilkoma aktywnościami. Przykład znajdziesz na następnej stronie.



# Zadanie z przenoszeniem danych pomiędzy aktywnościami

```
package com.example.simpleapp;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private EditText editName;
    private Button buttonNext;

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

        editName = findViewById(R.id.editName);
        buttonNext = findViewById(R.id.buttonNext);

        buttonNext.setOnClickListener(v -> {
            String name = editName.getText().toString().trim();

            Intent intent = new Intent(MainActivity.this, SecondActivity.class);
            intent.putExtra("USER_NAME", name);
            startActivity(intent);
        });
    }
}
```

```
package com.example.simpleapp;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private EditText editName;
    private Button buttonNext;

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

        editName = findViewById(R.id.editName);
        buttonNext = findViewById(R.id.buttonNext);

        buttonNext.setOnClickListener(v -> {
            String name = editName.getText().toString().trim();

            Intent intent = new Intent(MainActivity.this, SecondActivity.class);
            intent.putExtra("USER_NAME", name);
            startActivity(intent);
        });
    }
}
```

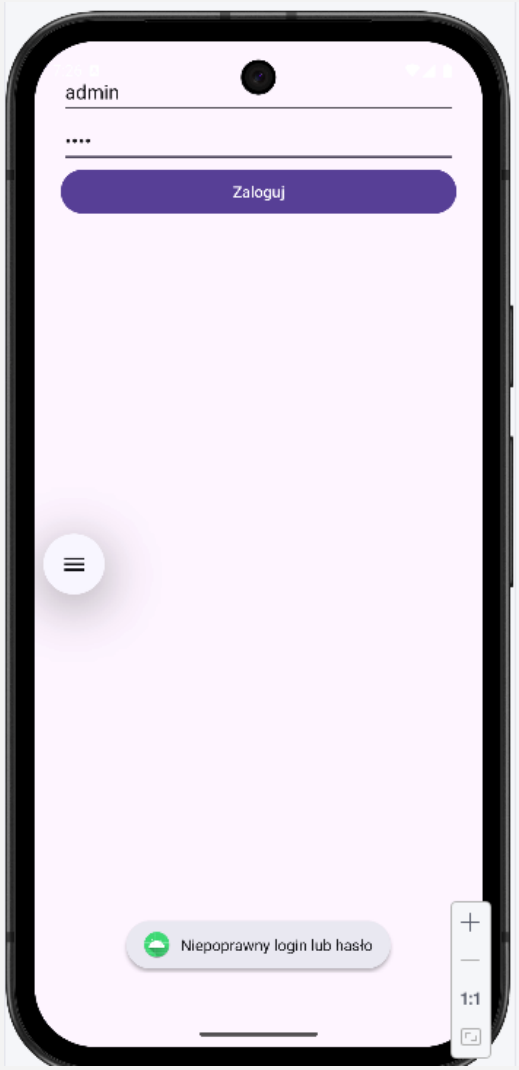
# Zadanie z przenoszeniem danych pomiędzy aktywnościami

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.simpleapp">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MaterialComponents">

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

        <activity android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

    </application>

</manifest>
```

# WEB API? Spokojnie, ale warto znać

```
private void fetchWeather() {
    String city = etCity.getText().toString().trim();

    if (city.isEmpty()) {
        tvWeather.setText("Podaj miasto!");
        return;
    }

    String url = "https://api.openweathermap.org/data/2.5/weather?q=" +
city +
            "&appid=" + API_KEY + "&units=metric&lang=pl";

    RequestQueue queue = Volley.newRequestQueue(this);

    StringRequest stringRequest = new StringRequest(Request.Method.GET,
url,
            response -> {
                try {
                    JSONObject json = new JSONObject(response);
                    JSONObject main = json.getJSONObject("main");
                    String temp = main.getString("temp");

                    JSONObject weather =
json.getJSONArray("weather").getJSONObject(0);
                    String description = weather.getString("description");

                    String result = "Miasto: " + city + "\nTemperatura: " + temp +
"°C\nOpis: " + description;
                    tvWeather.setText(result);
                } catch (Exception e) {
                    tvWeather.setText("Błąd parsowania danych.");
                    Log.e("WeatherApp", "Parse error", e);
                }
            },
            error -> {
                tvWeather.setText("Błąd połączenia: " + error.getMessage());
                Log.e("WeatherApp", "Volley error", error);
            });

    queue.add(stringRequest);
}
```

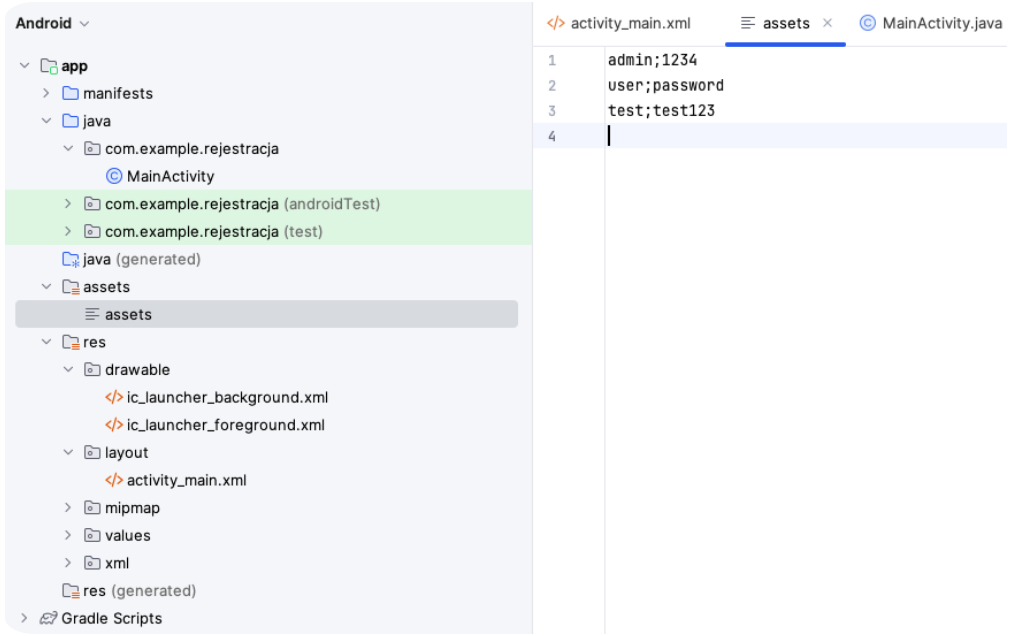
To po prostu ciąg znaków wygenerowanych w celu połączenia aplikacji z zewnętrznymi zasobami – najczęściej ze stroną internetową, która udostępnia np. pogodę lub aktualne wiadomości. Na egzaminie nie pojawi się z prostej przyczyny – nie można korzystać z internetu.




# Rejestracja użytkowników

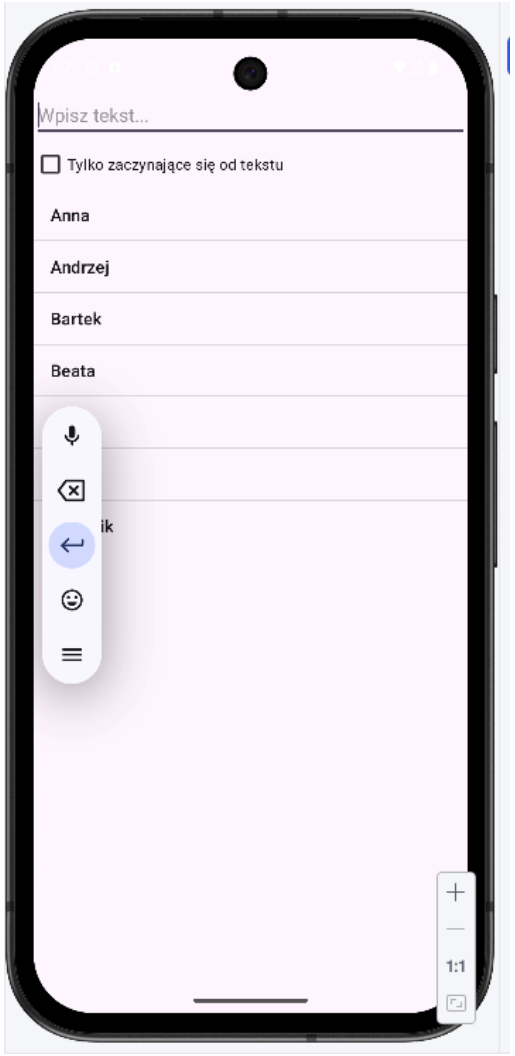
```
package com.example.rejestracja;

import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.HashMap;

public class MainActivity extends AppCompatActivity{

  EditText usernameInput, passwordInput;
  Button loginBtn;
  HashMap<String, String> users = new HashMap<>();

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

    usernameInput = findViewById(R.id.usernameInput);
    passwordInput = findViewById(R.id.passwordInput);
    loginBtn = findViewById(R.id.loginBtn);

    loadUsersFromFile();

    loginBtn.setOnClickListener(v -> {
      String username =
usernameInput.getText().toString().trim();
      String password =
passwordInput.getText().toString().trim();

      if (users.containsKey(username) &&
users.get(username).equals(password)) {
        Toast.makeText(this, "Zalogowano pomyślnie!",
Toast.LENGTH_SHORT).show();
      } else{
        Toast.makeText(this, "Niepoprawny login lub hasło",
Toast.LENGTH_SHORT).show();
      }
    });
  }

  private void loadUsersFromFile(){
    try{
      BufferedReader reader = new BufferedReader(
          new InputStreamReader(getAssets().open("users.txt"))
      );
      String line;
      while ((line = reader.readLine()) != null){
        String[] parts = line.split(";");
        if (parts.length == 2){
          users.put(parts[0].trim(), parts[1].trim());
        }
      }
      reader.close();
    } catch (Exception e){
      Toast.makeText(this, "Błąd wczytywania użytkowników",
Toast.LENGTH_LONG).show();
    }
  }
}
```

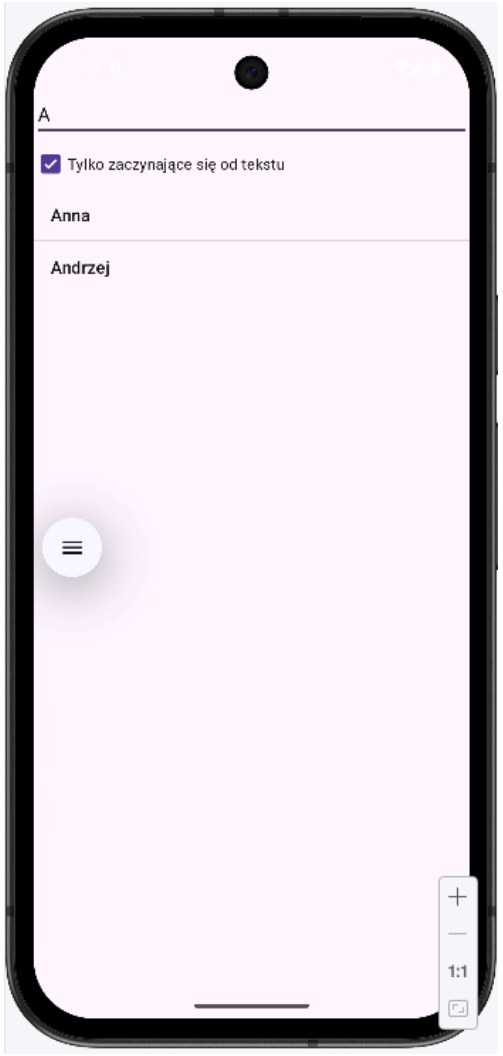
# Rejestracja użytkowników




```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:padding="24dp">

  <EditText
    android:id="@+id/usernameInput"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Nazwa użytkownika" />

  <EditText
    android:id="@+id/passwordInput"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Hasło"
    android:inputType="textPassword" />

  <Button
    android:id="@+id/loginBtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Zaloguj" />
</LinearLayout>
```

# Wyszukiwarka

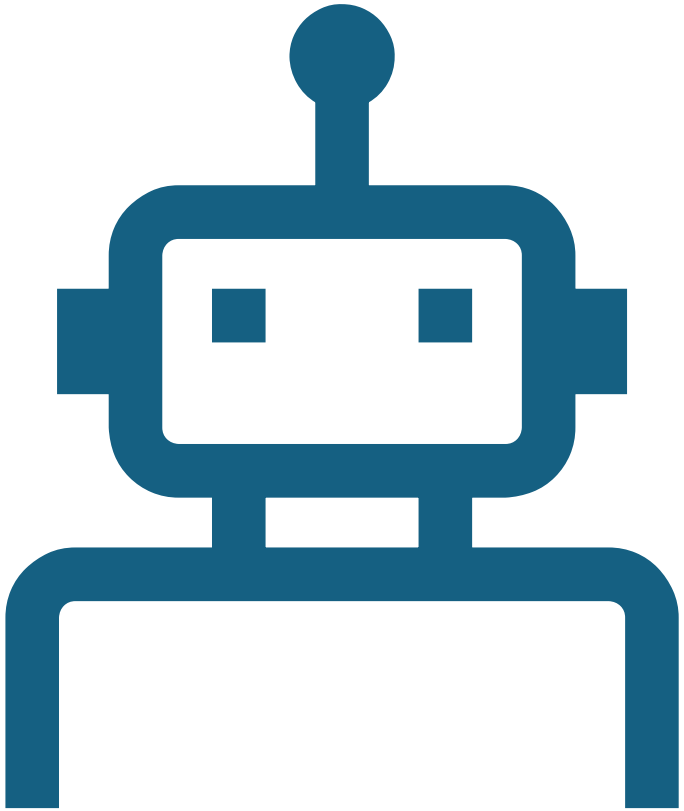



Aplikacja wyszukuje imiona z listy. Po zaznaczeniu pola wybiera tylko te imiona zaczynające się od liter (tak, trochę głupi przykład).

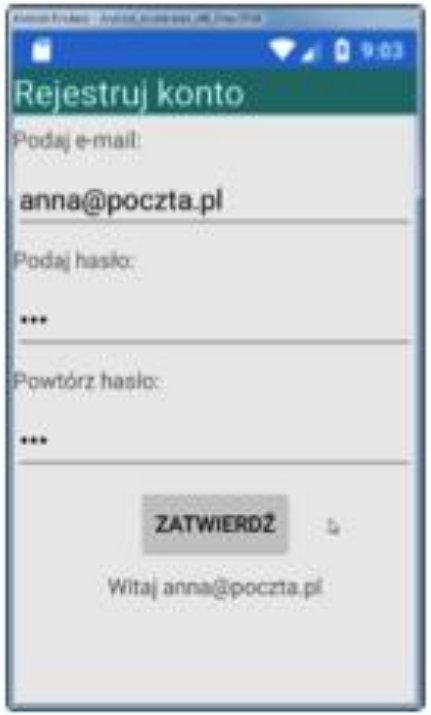
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/
android"
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingTop="50dp">

  <EditText
    android:id="@+id/searchEditText"
    android:hint="Wpisz tekst..."
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

  <CheckBox
    android:id="@+id/startsWithCheckbox"
    android:text="Tylko zaczynające się od tekstu"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

  <ListView
    android:id="@+id/resultListView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

</LinearLayout>
```

# Wyszukiwarka



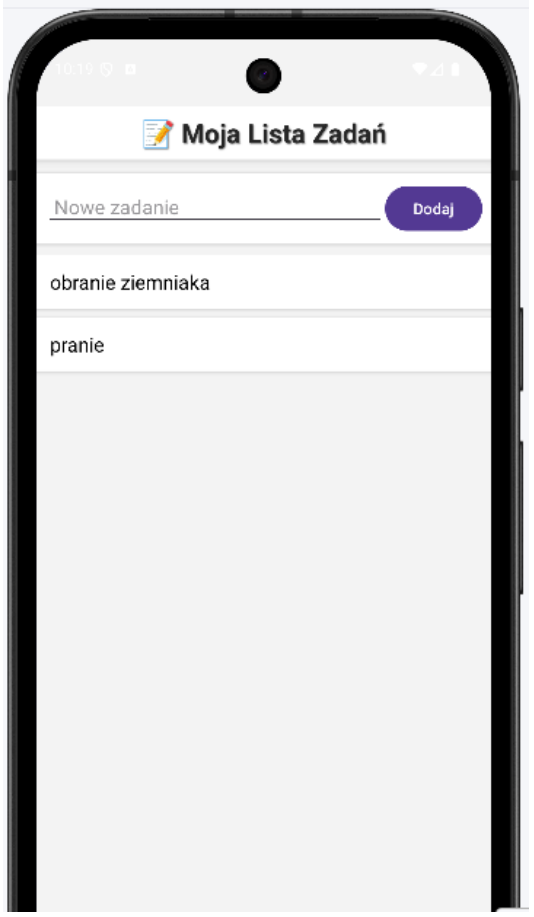
```
package com.example.wyszukaj;

import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.widget.ArrayAdapter;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.ListView;
import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity{

    EditText searchEditText;
    CheckBox startsWithCheckbox;
    ListView resultListView;

    List<String> itemList;
    ArrayAdapter<String> adapter;

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

        searchEditText = findViewById(R.id.searchEditText);
        startsWithCheckbox =
findViewById(R.id.startsWithCheckbox);
        resultListView = findViewById(R.id.resultListView);

        // Przykładowe dane
        itemList = new ArrayList<>();
        itemList.add("Anna");
        itemList.add("Andrzej");
        itemList.add("Bartek");
        itemList.add("Beata");
        itemList.add("Celina");
        itemList.add("Daniel");
        itemList.add("Dominik");

        adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, new ArrayList<>(itemList));
        resultListView.setAdapter(adapter);

        // Obsługa zmian tekstu
        searchEditText.addTextChangedListener(new TextWatcher()
{
            @Override public void beforeTextChanged(CharSequence
s, int start, int count, int after) {}
            @Override public void afterTextChanged(Editable s) {}

            @Override
            public void onTextChanged(CharSequence s, int start, int
before, int count) {
                filterList(s.toString(), startsWithCheckbox.isChecked());
            }
        });

        // Obsługa zaznaczenia checkboxa

startsWithCheckbox.setOnCheckedChangeListener((buttonVie
w, isChecked) -> {
            filterList(searchEditText.getText().toString(), isChecked);
        });
    }

    private void filterList(String query, boolean startsWith) {
        List<String> filtered = new ArrayList<>();
        for (String item : itemList) {
            if (startsWith) {
                if (item.toLowerCase().startsWith(query.toLowerCase()))
{
                    filtered.add(item);
                }
            } else {
                if (item.toLowerCase().contains(query.toLowerCase())) {
                    filtered.add(item);
                }
            }
        }
        adapter.clear();
        adapter.addAll(filtered);
        adapter.notifyDataSetChanged();
    }
}
```



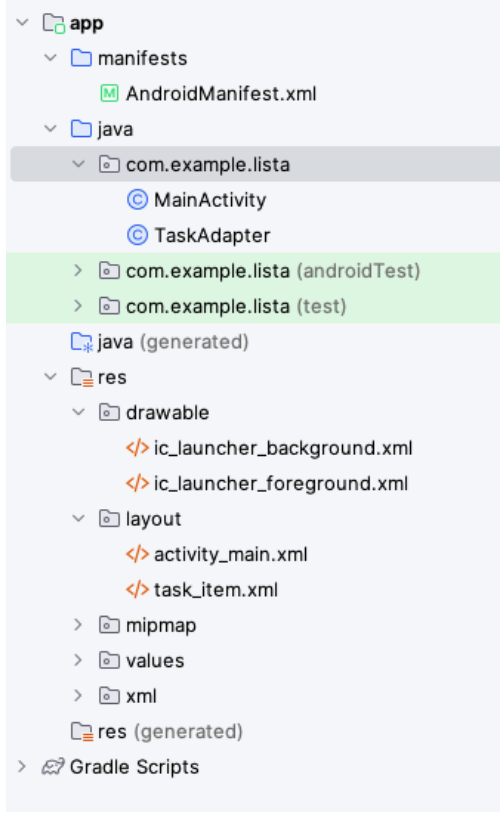




# Stare arkusze

# „ToDoLista” – popularny przykład




```
package com.example.lista;

import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;
import
androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

public class MainActivity extends AppCompatActivity{

    private EditText editTask;
    private Button btnAdd;
    private RecyclerView recyclerTasks;
    private List<String> taskList;
    private TaskAdapter adapter;
    private SharedPreferences prefs;

    private static final String PREFS_NAME = "MyTasks";
    private static final String KEY_TASKS = "tasks";

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

        editTask = findViewById(R.id.editTask);
        btnAdd = findViewById(R.id.btnAdd);
        recyclerTasks = findViewById(R.id.recyclerTasks);

        prefs = getSharedPreferences(PREFS_NAME,
MODE_PRIVATE);
        taskList = new ArrayList<>(loadTasks());

        adapter = new TaskAdapter(taskList, position -> {
            taskList.remove(position);
            saveTasks();
            adapter.notifyItemRemoved(position);
        });

        recyclerTasks.setLayoutManager(new
LinearLayoutManager(this));
        recyclerTasks.setAdapter(adapter);

        btnAdd.setOnClickListener(v -> {
            String task = editTask.getText().toString().trim();
            if (!task.isEmpty()) {
                taskList.add(task);
                saveTasks();
                adapter.notifyItemInserted(taskList.size() - 1);
                editTask.setText("");
            }
        });
    }

    private Set<String> loadTasks() {
        return prefs.getStringSet(KEY_TASKS, new
HashSet<>());
    }

    private void saveTasks() {
        prefs.edit().putStringSet(KEY_TASKS, new
HashSet<>(taskList)).apply();
    }
}
```



# „ToDoLista” – popularny przykład

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

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

  android:layout_width="match_parent"

  android:layout_height="match_parent"

  android:orientation="vertical"

  android:background="#F5F5F5"

  android:padding="16dp">


  <!-- Nagłówek -->

  <TextView

    android:id="@+id/tvHeader"

    android:layout_width="match_parent"

    android:layout_height="wrap_content"

    android:text="📝 Moja Lista Zadań"

    android:textSize="24sp"

    android:textStyle="bold"

    android:textColor="#333333"

    android:gravity="center"

    android:padding="8dp"

    android:background="#FFFFFF"

    android:elevation="4dp"

    android:layout_marginBottom="12dp"

    android:shadowColor="#999999"

    android:shadowDx="2"

    android:shadowDy="2"

    android:shadowRadius="2" />


<!-- Pole dodawania zadania -->

<LinearLayout

  android:layout_width="match_parent"

  android:layout_height="wrap_content"

  android:orientation="horizontal"

  android:background="#FFFFFF"

  android:padding="8dp"

  android:elevation="2dp"

  android:layout_marginBottom="10dp">


  <EditText

    android:id="@+id/editTask"

    android:layout_width="0dp"

    android:layout_weight="1"

    android:layout_height="wrap_content"

    android:hint="Nowe zadanie"

    android:padding="8dp" />


  <Button

    android:id="@+id/btnAdd"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:text="Dodaj" />

</LinearLayout>


<!-- Lista zadań -->

<androidx.recyclerview.widget.RecyclerView

  android:id="@+id/recyclerTasks"

  android:layout_width="match_parent"

  android:layout_height="match_parent" />


</LinearLayout>
```

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

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

android:layout_width="match_parent"

android:layout_height="wrap_content"

  android:orientation="horizontal"

  android:background="#FFFFFF"

  android:padding="12dp"

  android:layout_marginBottom="8dp"

  android:elevation="2dp">


  <TextView

    android:id="@+id/tvTask"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:text="Przykładowe zadanie"

    android:textSize="18sp"

    android:textColor="#000000" />


</LinearLayout>
```

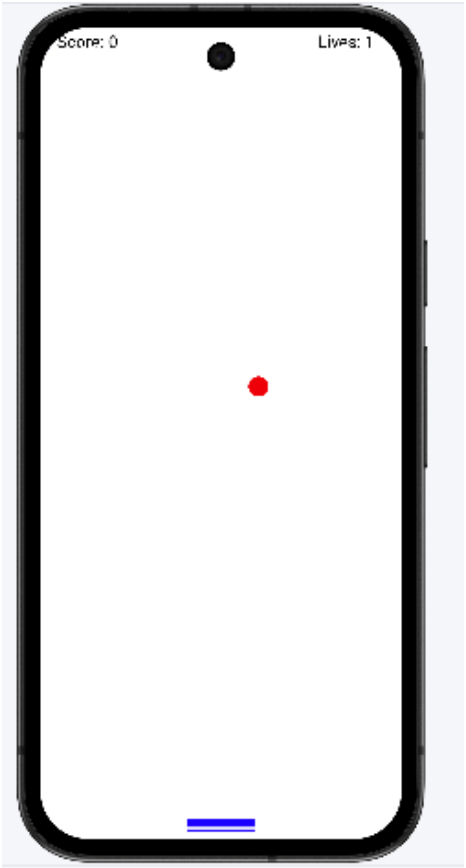
# „ToDoLista” – popularny przykład

```
package com.example.todolist;


import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.TextView;


import androidx.annotation.NonNull;

import
androidx.recyclerview.widget.RecyclerV
iew;


import java.util.List;


public class TaskAdapter extends
RecyclerView.Adapter<TaskAdapter.Tas
kViewHolder> {


   private List<String> tasks;

   private OnTaskClickListener listener;


   public interface OnTaskClickListener{

      void onTaskClick(int position);

   }
```

```
   public TaskAdapter(List<String> tasks,
OnTaskClickListener listener) {

      this.tasks = tasks;

      this.listener = listener;

   }


   @NonNull

   @Override

   public TaskViewHolder
onCreateViewHolder(@NonNull
ViewGroup parent, int viewType) {

      View view =
LayoutInflater.from(parent.getContext())

            .inflate(R.layout.task_item,
parent, false);

      return new TaskViewHolder(view);

   }


   @Override

   public void
onBindViewHolder(@NonNull
TaskViewHolder holder, int position) {


holder.tvTask.setText(tasks.get(position
));

   }
```

```
   @Override

   public int getItemCount() {

      return tasks.size();

   }


   class TaskViewHolder extends
RecyclerView.ViewHolder {

      TextView tvTask;


      TaskViewHolder(View itemView) {

         super(itemView);

         tvTask =
itemView.findViewById(R.id.tvTask);


         itemView.setOnClickListener(v ->
listener.onTaskClick(getAdapterPosition
()));

      }

   }

}
```

# Gra „Złap piłkę”




```
package com.example.catchtheball;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

public class GameView extends SurfaceView
implements SurfaceHolder.Callback {

  private GameThread thread;
  private Paint paint;
  private float basketX;
  private float basketWidth = 200;
  private float basketHeight = 40;

  private float ballX, ballY;
  private float ballRadius = 30;
  private float ballSpeed = 8;

  private int score = 0;
  private int lives = 3;

  public GameView(Context context) {
    super(context);
    getHolder().addCallback(this);
    paint = new Paint();
    setFocusable(true);
    resetBall();
  }

  private void resetBall() {
    ballX = (float) (Math.random() * (getWidth() -
ballRadius * 2)) + ballRadius;
    ballY = 0;
  }

  @Override
  public void surfaceCreated(SurfaceHolder
holder) {
    basketX = getWidth() / 2f - basketWidth / 2;
    thread = new GameThread(getHolder());
    thread.setRunning(true);
    thread.start();
  }

  @Override
  public void surfaceChanged(SurfaceHolder
holder, int format, int width, int height) {}

  @Override
  public void surfaceDestroyed(SurfaceHolder
holder) {
    boolean retry = true;
    thread.setRunning(false);
    while (retry) {
      try {
        thread.join();
        retry = false;
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
    }
  }

  @Override
  public boolean onTouchEvent(MotionEvent
event) {
    if (event.getAction() ==
MotionEvent.ACTION_MOVE) {
      basketX = event.getX() - basketWidth / 2;
    }
    return true;
  }

  public void update() {
    ballY += ballSpeed;

    if (ballY + ballRadius > getHeight() -
basketHeight &&
        ballX > basketX && ballX < basketX +
basketWidth) {
      score++;
      resetBall();
    }

    if (ballY > getHeight()) {
      lives--;
      resetBall();
      if (lives <= 0) {
        score = 0;
        lives = 3;
      }
    }
  }

  @Override
  public void draw(Canvas canvas) {
    super.draw(canvas);
    if (canvas != null) {
      canvas.drawColor(Color.WHITE);

      paint.setColor(Color.BLUE);
      canvas.drawRect(basketX, getHeight() -
basketHeight - 20,
          basketX + basketWidth, getHeight() -
20, paint);

      paint.setColor(Color.RED);
      canvas.drawCircle(ballX, ballY, ballRadius,
paint);

      paint.setColor(Color.BLACK);
      paint.setTextSize(50);
      canvas.drawText("Score: " + score, 50, 60,
paint);
      canvas.drawText("Lives: " + lives,
getWidth() - 250, 60, paint);
    }
  }

  class GameThread extends Thread {
    private SurfaceHolder surfaceHolder;
    private boolean running;

    public GameThread(SurfaceHolder holder) {
      surfaceHolder = holder;
    }

    public void setRunning(boolean run) {
      running = run;
    }

    @Override
    public void run() {
      while (running) {
        Canvas canvas = null;
        try {
          canvas = surfaceHolder.lockCanvas();
          synchronized (surfaceHolder) {
            update();
            draw(canvas);
          }
        } finally {
          if (canvas != null) {

surfaceHolder.unlockCanvasAndPost(canvas);
          }
        }
        try {
          sleep(16); //~60 FPS
        } catch (InterruptedException e) {
          e.printStackTrace();
        }
      }
    }
  }
}
```

# Koniec?

Prezentacja zawiera najważniejsze elementy podstawy programowej. Aplikację mobilną możesz również napisać w języku Kotlin za pomocą Android Studio lub za pomocą C# oraz dedykowanego środowiska NET. Te zagadnienia znajdziesz w innych prezentacjach.