

1. Activity

- 1.1 Tworzenie nowej Activity
- 1.2 Komunikacja między Activity

2. Widżety interfejsu użytkownika

- 2.1 Przycisk (Button)
- 2.2 Pole tekstowe (EditText)
- 2.3 Lista (ListView)
- 2.4 Seekbar
- 2.5 Select

3. Interakcje użytkownika

- 3.1 Obsługa kliknięć (OnClickListener)
- 3.2 Długie kliknięcia (OnLongClickListener)
- 3.3 Dotyk (TouchEvent)
- 3.4 Powiadomienia (Toast, Snackbar)

4. Zarządzanie zasobami

- 4.1 Typy zasobów: colors

5. Funkcja na stringach i liczbach

1. Activity

1.1 Utworzenie nowego activity

1. Utwórz nowe activity

```
// SecondActivity.java
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;

public class SecondActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second); // Ustawienie layoutu dla tej Activity
    }
}
```

2. Utwórz nowy layout XML dla SecondActivity:

```

<?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:gravity="center">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello from SecondActivity!"
        android:textSize="24sp"/>
</LinearLayout>

```

3. Otwórz plik AndroidManifest.xml i dodaj nową Activity.

```

<application
    ... >
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name=".SecondActivity" />
</application>

```

4. Dodaj przycisk do layoutu activity_main.xml:

```
<?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:gravity="center">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Open Second Activity"/>
</LinearLayout>
```

5. Ustaw OnClickListener w MainActivity do uruchomienia SecondActivity:

```
import android.content.Intent;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;

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

        Button button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, SecondActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

1.2 Komunikacja między Activity -

Komunikacja między Activity odbywa się głównie za pomocą Intencji (Intent). Możemy przekazywać dane z jednej Activity do drugiej za pomocą `putExtra` i odczytywać je za pomocą `getIntent().getExtras()`.

1. Wysłanie danych z jednej Activity do drugiej:

```
Intent intent = new Intent(MainActivity.this, SecondActivity.class);
intent.putExtra("EXTRA_MESSAGE", "Hello from MainActivity!");
startActivity(intent);
```

2. Odbieranie danych w drugiej Activity:

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

        Intent intent = getIntent();
        String message = intent.getStringExtra("EXTRA_MESSAGE");
        TextView textView = findViewById(R.id.textview);
        textView.setText(message);
    }
}
```

2. Widżety interfejsu użytkownika

2.1 Przycisk (Button) - Przycisk umożliwia użytkownikowi wykonywanie akcji po jego kliknięciu.

- XML

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click Me"/>
```

- JAVA

```
Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(MainActivity.this, "Button clicked!", Toast.LENGTH_SHORT).show();
    }
});
```

2.2 Pole tekstowe (EditText)

- XML

```
<EditText
    android:id="@+id/editText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter text here"/>
```

- JAVA

```

EditText editText = findViewById(R.id.editText);
Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String inputText = editText.getText().toString();
        Toast.makeText(MainActivity.this, "You entered: " + inputText, Toast.LENGTH_SHORT).show();
    }
});

```

2.2 Lista (ListView)

- XML:

```

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

```

- JAVA:

```

ListView listView = findViewById(R.id.listView);
String[] items = {"Item 1", "Item 2", "Item 3", "Item 4"};
ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1,
listView.setAdapter(adapter);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        String selectedItem = (String) parent.getItemAtPosition(position);
        Toast.makeText(MainActivity.this, "You selected: " + selectedItem, Toast.LENGTH_SHOR
    }
});

```

2.3 SeekBar

- XML:

```

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

```

- JAVA:


```

SeekBar seekBar = findViewById(R.id.seekBar);
TextView textView = findViewById(R.id.textView);

seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
        textView.setText("Progress: " + progress);
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
        // Do something when tracking starts
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        // Do something when tracking stops
    }
});

```

2.4 Spinner (Select)

- XML:

```

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

```

- JAVA:

```

Spinner spinner = findViewById(R.id.spinner);
String[] items = {"Option 1", "Option 2", "Option 3"};
ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item, items);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinner.setAdapter(adapter);

spinner.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        String selectedItem = (String) parent.getItemAtPosition(position);
        Toast.makeText(MainActivity.this, "Selected: " + selectedItem, Toast.LENGTH_SHORT)
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {
        // Do something when nothing is selected
    }
});

```

3. Interakcje użytkownika w Androidzie

- 3.1 Obsługa kliknięć (OnClickListener)

```

Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(MainActivity.this, "Button clicked!", Toast.LENGTH_SHORT).show();
    }
});

```

- 3.2 Długie kliknięcia (OnLongClickListener)

```

Button button = findViewById(R.id.button);
button.setOnLongClickListener(new View.OnLongClickListener() {
    @Override
    public boolean onLongClick(View v) {
        Toast.makeText(MainActivity.this, "Button long clicked!", Toast.LENGTH_SHORT).show();
        return true; // true jeśli zdarzenie zostało obsłużone, false jeśli nie
    }
});

```

- **3.3 Dotyk i gesty (TouchEvent)**

```

TextView textView = findViewById(R.id.textView);
textView.setOnTouchListener(new View.OnTouchListener() {
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        return gestureDetector.onTouchEvent(event);
    }
});

```

- **3.4 Snackbar**

```

Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Snackbar.make(v, "This is a Snackbar", Snackbar.LENGTH_LONG)
            .setAction("Undo", new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Toast.makeText(MainActivity.this, "Undo clicked", Toast.LENGTH_SHORT)
                        .show();
                }
            })
            .show();
    }
});

```

4. Zarządzanie zasobami w Androidzie - Zarządzanie zasobami (resources) w Androidzie pozwala na oddzielenie danych takich jak obrazy, teksty, kolory i wymiary od kodu aplikacji. Dzięki temu aplikacje są bardziej elastyczne i łatwiejsze do utrzymania.

- **4.1 colors**

```
<resources>
    <color name="primaryColor">#3F51B5</color>
    <color name="primaryDarkColor">#303F9F</color>
    <color name="accentColor">#FF4081</color>
</resources>
```

5. Funkcje na stringach i liczbach

String Functions

Funkcja	Opis	Przed	Po
<code>length()</code>	Zwraca długość stringa	<code>"Hello"</code>	<code>5</code>
<code>charAt(int index)</code>	Zwraca znak na podanej pozycji	<code>"Hello"</code>	<code>'e'</code> (dla <code>charAt(1)</code>)
<code>substring(int beginIndex, int endIndex)</code>	Zwraca podciąg stringa	<code>"Hello"</code>	<code>"ell"</code> (dla <code>substring(1, 4)</code>)
<code>toUpperCase()</code>	Zamienia wszystkie znaki na wielkie litery	<code>"Hello"</code>	<code>"HELLO"</code>
<code>toLowerCase()</code>	Zamienia wszystkie znaki na małe litery	<code>"Hello"</code>	<code>"hello"</code>
<code>trim()</code>	Usuwa wiodące i końcowe białe znaki	<code>" Hello "</code>	<code>"Hello"</code>
<code>replace(CharSequence target, CharSequence replacement)</code>	Zastępuje wszystkie wystąpienia podciągu	<code>"Hello"</code>	<code>"Hezzo"</code> (dla <code>replace("l", "z")</code>)
<code>contains(CharSequence s)</code>	Sprawdza, czy string zawiera podciąg	<code>"Hello"</code>	<code>true</code> (dla <code>contains("ell")</code>)
<code>equals(Object anObject)</code>	Sprawdza, czy dwa stringi są równe	<code>"Hello"</code> i <code>"Hello"</code>	<code>true</code>
<code>equalsIgnoreCase(String anotherString)</code>	Sprawdza, czy dwa stringi są równe, ignorując wielkość liter	<code>"Hello"</code> i <code>"hello"</code>	<code>true</code>
<code>split(String regex)</code>	Dzieli stringa na tablicę podciągów ↓	<code>"a,b,c"</code>	<code>["a", "b", "c"]</code> (dla <code>split(",")</code>)

Number Functions

Funkcja	Opis	Przed	Po
<code>parseInt(String s)</code>	Konwertuje string na <code>int</code>	<code>"123"</code>	<code>123</code>
<code>parseFloat(String s)</code>	Konwertuje string na <code>double</code>	<code>"123.45"</code>	<code>123.45</code>
<code>parseFloat(String s)</code>	Konwertuje string na <code>float</code>	<code>"123.45"</code>	<code>123.45f</code>
<code>parseLong(String s)</code>	Konwertuje string na <code>long</code>	<code>"123456789"</code>	<code>123456789L</code>
<code>toString()</code>	Konwertuje liczbę na string	<code>123</code>	<code>"123"</code>
<code>intValue()</code>	Zwraca wartość liczbową jako <code>int</code>	<code>Double.valueOf(123.45)</code>	<code>123</code>
<code>doubleValue()</code>	Zwraca wartość liczbową jako <code>double</code>	<code>Float.valueOf(123.45f)</code>	<code>123.45</code>
<code>compareTo(Integer anotherInteger)</code>	Porównuje dwie liczby	<code>10</code> i <code>20</code>	<code>-1</code> (dla <code>10.compareTo(20)</code>)
<code>max(int a, int b)</code>	Zwraca większą z dwóch liczb	<code>10</code> i <code>20</code>	<code>20</code>
<code>min(int a, int b)</code>	Zwraca mniejszą z dwóch liczb	<code>10</code> i <code>20</code>	<code>10</code>
<code>Math.round(float a)</code>	Zaokrągla liczbę do najbliższej wartości całkowitej	<code>123.45f</code>	<code>123</code>
<code>Math.ceil(double a)</code>	Zaokrągla liczbę w górę do najbliższej wartości całkowitej	<code>123.45</code>	<code>124</code>
<code>Math.floor(double a)</code>	Zaokrągla liczbę w dół do najbliższej wartości całkowitej	<code>123.45</code>	<code>123</code>

an