PROGRAMOWANIE APLIKACJI MOBILNYCH

AKTYWNOŚCI I INTENCJE

dr Artur Bartoszewski

Intencje

- ✓ Intencje (obok Aktywności) są jednym z podstawowych komponentów z których zbudowane są aplikacje systemu Android.
- ✓ Są one odpowiedzialne przede wszystkim za obsługę rozkazów wydawanych przez użytkownika.
- ✓ Za pomocą intencji możemy wprowadzić komunikację pomiędzy aplikacjami (lub mniejszymi komponentami, jak usługi, aktywności itp.).
- ✓ Jednak najważniejszym zadaniem tego komponentu jest uruchamianie aplikacji lub aktywności.

Uruchamianie aktywności

Jawne (explicit) – w których wskazujemy obiekt, który chcemy stworzyć. W tym wypadku jednym z argumentów konstruktora Intencji jest obiekt typu Class wskazujący na klasę, której obiekt chcemy stworzyć.

Na przykład:

```
Intent intent = new Intent(context, MainActivity.class);
```

Tak zdefiniowana intencja uruchomi aktywność MainActivity.

Uruchamianie aktywności

Niejawne (implicit) – w których zawarta jest informacje o tym co chcemy zrobić, bez podawania konkretnych klas, które mają to zrealizować.

Najczęściej podawane są dwie informacje:

- co chcemy zrobić
- na jakich danych chcemy tą czynność wykonać.

System, za pomocą Filtrów Intencji, (o których wspominaliśmy na 1 wykładzie) decyduje jaka Aktywność ma być uruchomiona.

Na przykład:

```
Intent = new Intent ( Intent.ACTION_VIEW,
                      Uri.parse("http://www.google.com") );
```

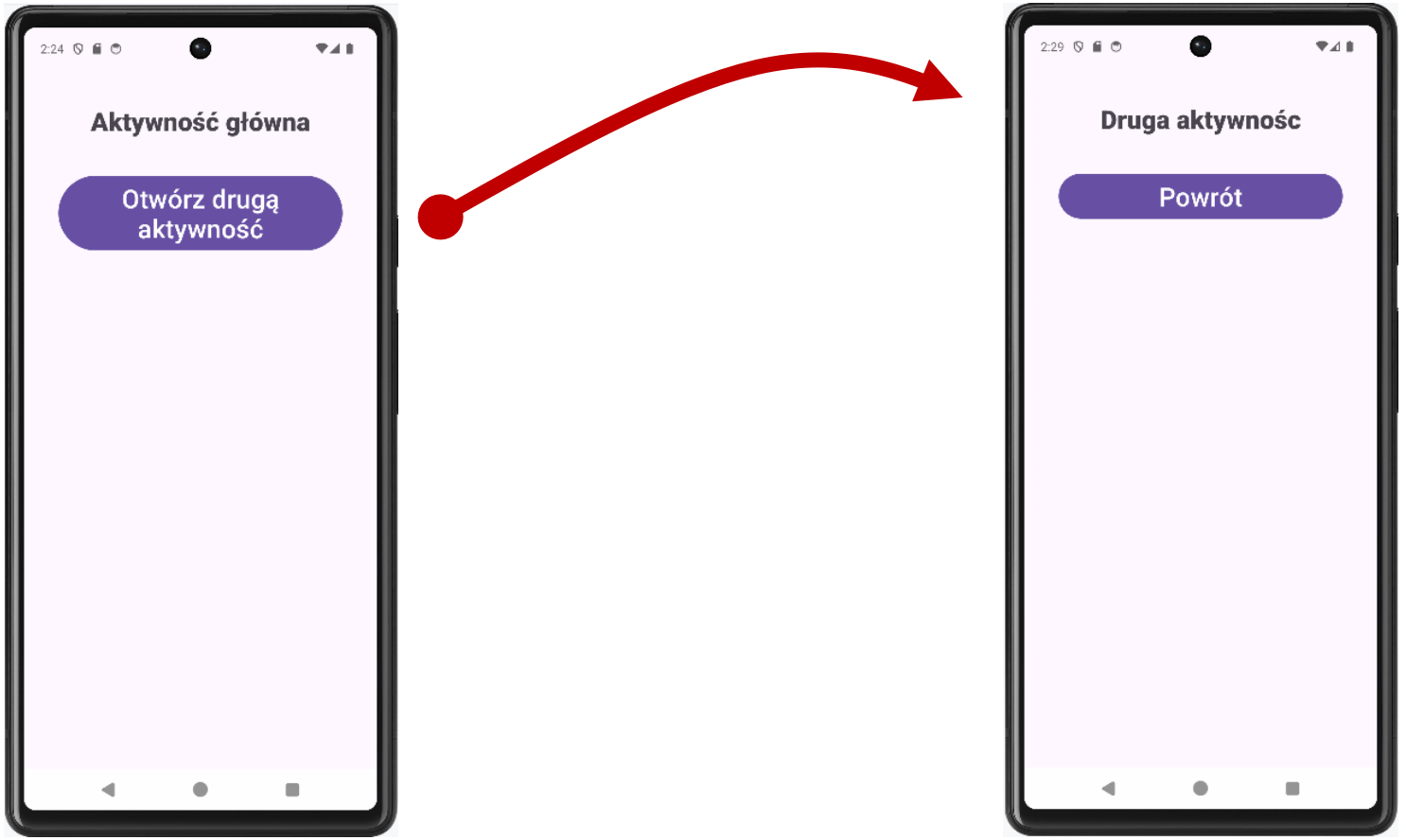
Informujemy system o tym, że chcemy zobaczyć dane (Intent.ACTION_VIEW) zapisane pod adresem URI

Aplikacje mobilne

Tworzymy aplikację, która zaprezentuje 5 różnych typów wykorzystania intencji do komunikacje pomiędzy aktywnościami

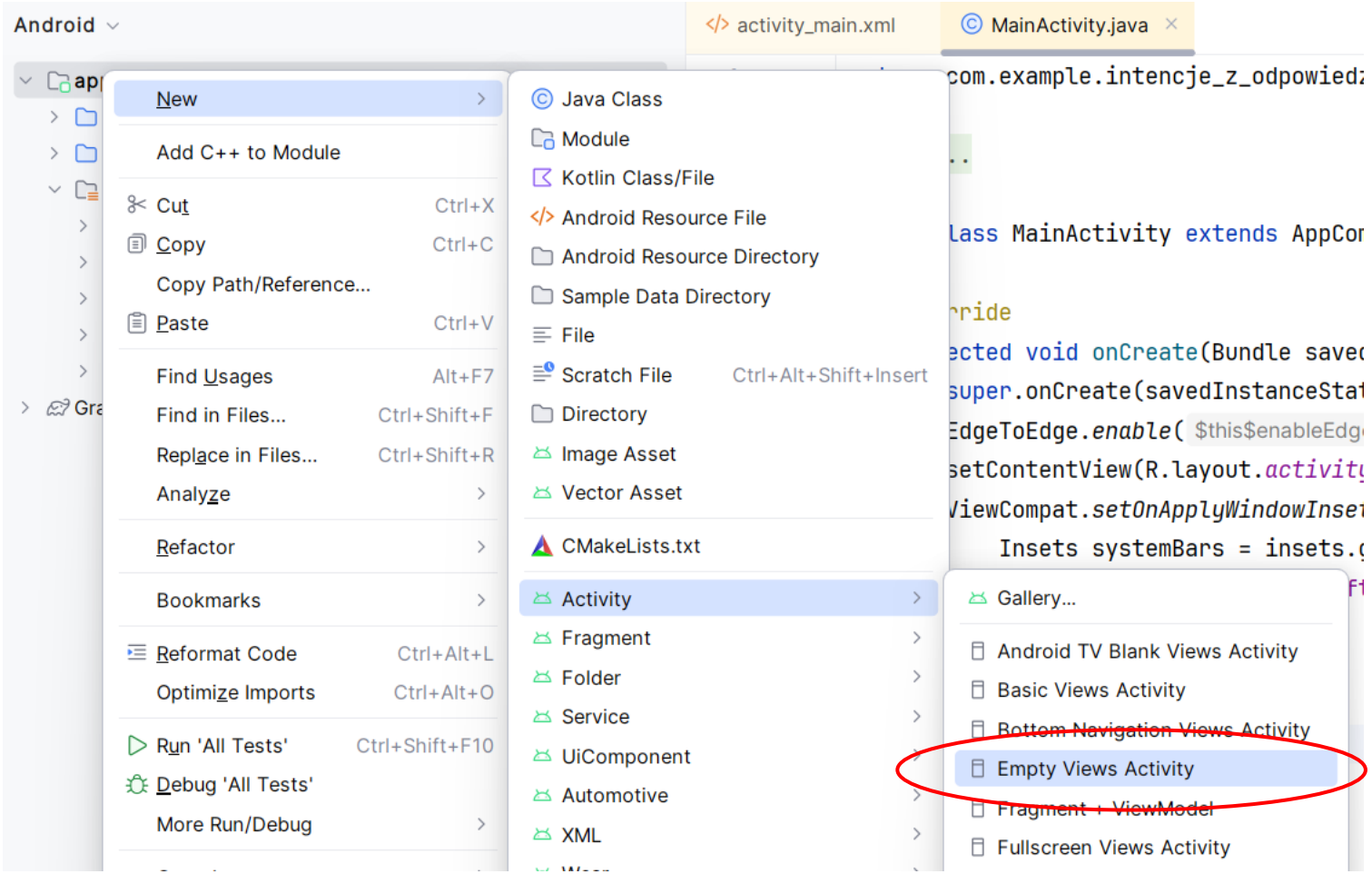
1. Otwieranie aktywności,
2. Otwieranie aktywności i jednoczesne przekazywanie do niej danych,
3. Otwieranie aktywności i odbieranie od niej odpowiedzi;
4. Obsługa adresów WWW;
5. Obsługa adresów geoUri

Otwieranie nowej aktywności



Otwieranie nowej aktywności

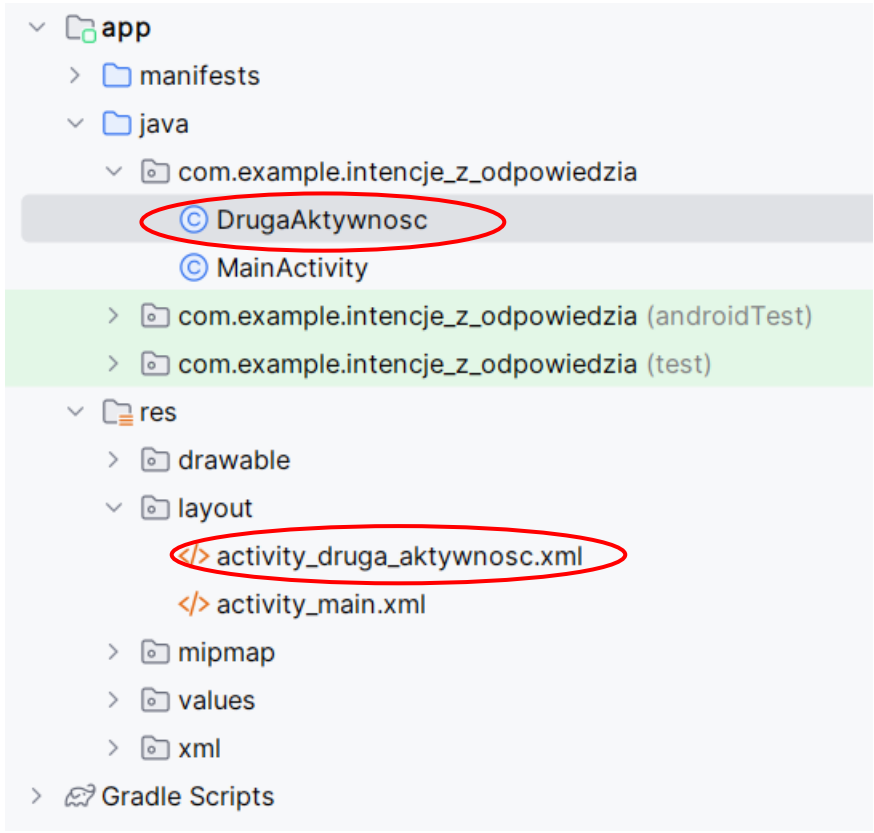
Jeżeli otwierać będziemy aktywność własnej aplikacji (intencja typu explicit), rozpoczynamy od dodania do projektu nowej aktywności.



Otwieranie nowej aktywności

Uzyskujemy:

1. Plik .java
2. Powiązany z nim plik .xml



Otwieranie nowej aktywności

W aktywności głównej przygotowujemy intencje która poinformuję system że należy otworzyć wskazaną aktywność.

- Pierwszym parametrem jest kontekst, który w tym wypadku stanowi też adres powrotu, Gdyż po zamknięciu drugiej aktywności system nie zamyka całej aplikacji lecz wraca do aktywności która ją wywołała.
  - Jako kontekst możemy użyć wskaźnika this lub getApplicationContext()
- drugim parametrem jest nazwa klasy odpowiadającej za aktywność którą chcemy otworzyć

```
Intent intent = new Intent(getApplicationContext(), MainActivity2.class);
```

Tak przygotowaną intencję należy wysłać do systemu za pomocą polecenia:

```
startActivity(intent);
```

Otwieranie nowej aktywności

W aktywności głównej przygotowujemy intencje która poinformuję system że należy otworzyć wskazaną aktywność.

- Pierwszym parametrem jest kontekst, który w tym wypadku stanowi też adres powrotu, Gdyż po zamknięciu drugiej aktywności system nie zamyka całej aplikacji lecz wraca do aktywności która ją wywołała.
  - Jako kontekst możemy użyć wskaźnika this lub getApplicationContext()
- drugim parametrem jest nazwa klasy odpowiadającej za aktywność którą chcemy otworzyć

```
Intent intent = new Intent(getApplicationContext(), MainActivity2.class);
```

Tak przygotowaną intencję należy wysłać do systemu za pomocą polecenia:

```
startActivity(intent);
```

Otwieranie nowej aktywności

Przykład zastosowania:

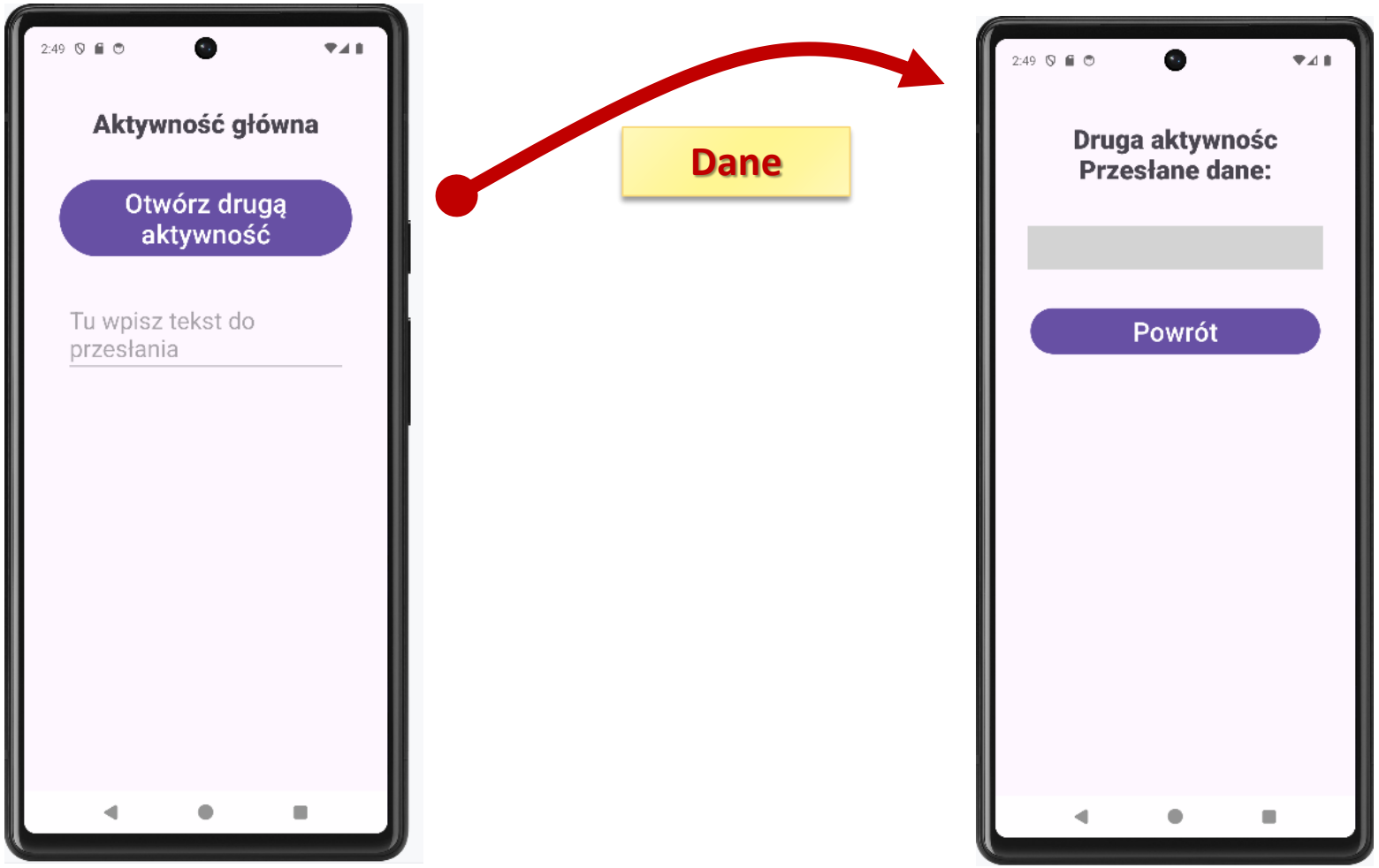
Otworzenie aktywności

```
Button button01 = findViewById(R.id.button01);
button01.setOnClickListener(v -> {
   Intent intent = new Intent(getApplicationContext(), MainActivity2.class);
   startActivity(intent);
});
```

Powrót do aktywności nadrzędnej
(w pliku nowej aktywności np.: MainActivity2.java)

```
Button button01 = findViewById(R.id.button03);
button01.setOnClickListener(v -> {
   finish();
});
```

Otwieranie aktywności i przesyłanie do niej danych



Otwieranie aktywności i przesyłanie do niej danych

Intencje nie tylko wywołują aktywności lecz także mogą przenosić pomiędzy nimi różnego typu dane.

Dane dołączyć należy do aktywności za pomocą metody putEkstra(). Przesyłane są one jako para klucz -> wartość

```
EditText editText = findViewById(R.id.editText01);
Button button01 = findViewById(R.id.button01);
button01.setOnClickListener(v -> {
    Intent intent = new Intent(getApplicationContext(), MainActivity2.class);
    intent.putExtra("text", editText.getText().toString());
    startActivity(intent);
});
```

Do jednej intencji dołączyć można wiele wartości wywołując wielokrotnie metodę putEkstra().

Otwieranie aktywności i przesyłanie do niej danych

W aktywności docelowej intencja odbierana jest przez metodę onCreate(), w której mamy dostęp do dołączonych do niej dodatków.

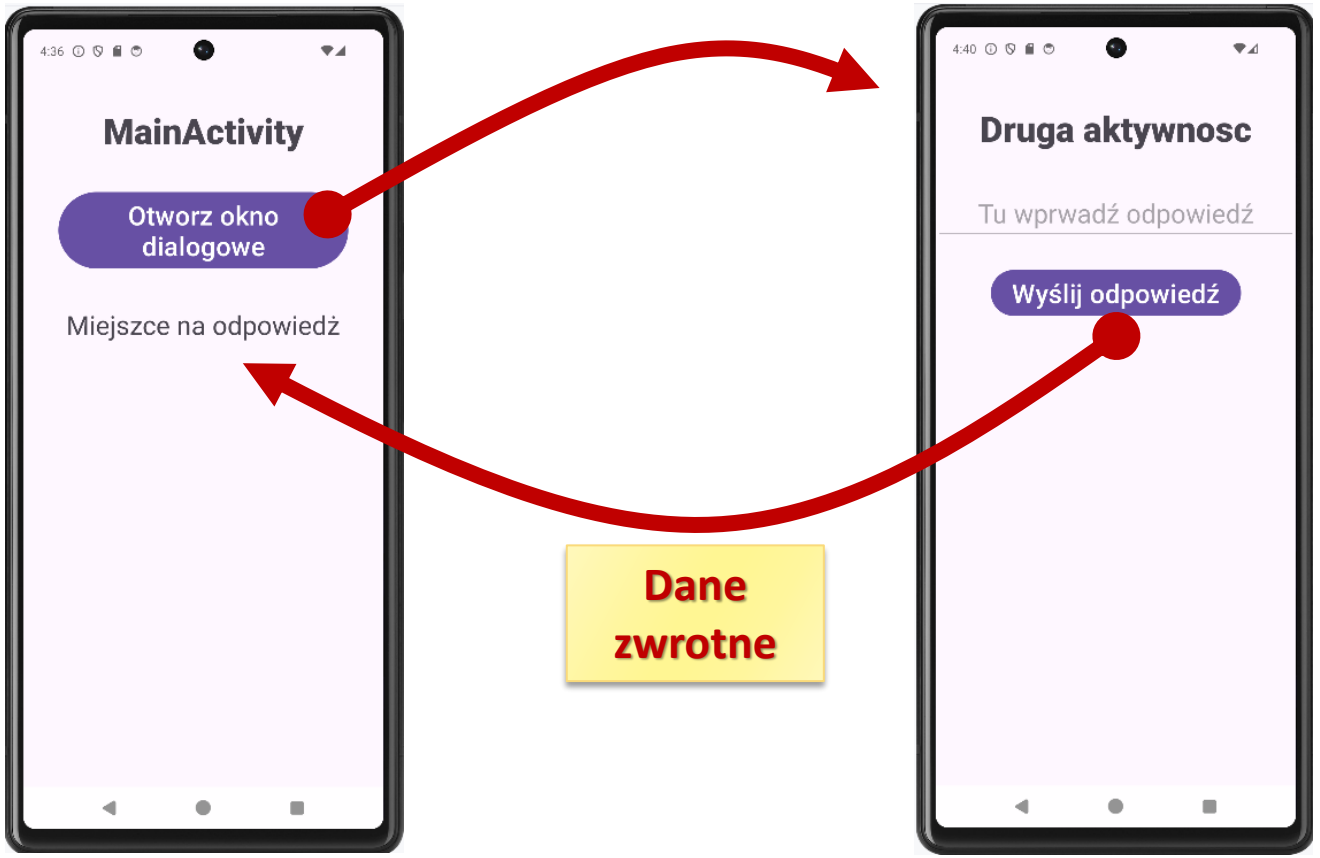
- Dodatki można zapisać w zmiennej typu Bundle (która przechowuje zbiór danych w parach KLUCZ -> WARTOŚĆ).
- Do wyciągnięcia danych z paczku musimy znać klucz.

```
TextView textView03 = findViewById(R.id.textView03);
Bundle extras = getIntent().getExtras();
String text = extras.getString("text");
textView03.setText(text);
```

Można też krócej, zmiennych pośredniczących:

```
textView03.setText(getIntent().getStringExtra("text"));
```

Otwieranie aktywności i odbieranie od niej odpowiedzi



Otwieranie aktywności i odbieranie od niej odpowiedzi

Od otwieranej aktywności często oczekujemy jakieś odpowiedzi – prze4słania danych z aktywności podrzędnej do tej, która ją uruchomiła. Przykładem może tu być tu okno dialogowe, czy też formularz do wprowadzania danych.

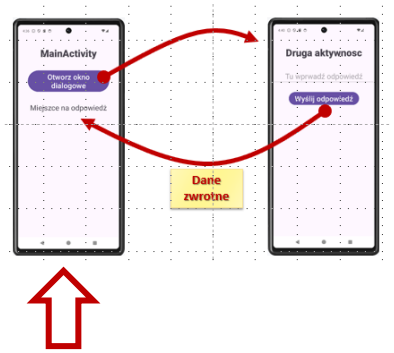
Aby otworzyć aktywność która zwróci nam jakieś dane posłużymy się obiektem ActivityResultLauncher. Jest to nowoczesny mechanizm do zarządzania wynikami aktywności, zastępujący starszą metodę startActivityForResult().

Aby móc korzystać z launchera należy dodać zależności dla biblioteki androidx.activity w pliku build.gradle (Module: app):

```
dependencies {

implementation("androidx.activity:activity:1.2.0")
    implementation(libs.appcompat)
    implementation(libs.material)
```

Otwieranie aktywności i odbieranie od niej odpowiedzi



1. Deklaracja referencji do zmiennej typu ActivityResultLauncher, która będzie służyła do uruchamiania aktywności i obsługiwania jej wyniku (Najlepiej globalnie, jako pole klasy MainActivity)

```
private ActivityResultLauncher<Intent> activityLauncher;
```

2. Rejestracja launchera (najczęściej robimy to w metodzie onCreate)

```
activityLauncher = registerForActivityResult(
        new ActivityResultContracts.StartActivityForResult(),
        new ActivityResultCallback<ActivityResult>() {
            @Override
            public void onActivityResult(ActivityResult o) {
                //reakcja na odpowiedź aktywności
            }
        });
```

Otwieranie aktywności i odbieranie od niej odpowiedzi

Funkcja registerForActivityResult() – potrzebuje 2 argumentów będących obiektami.

1. ActivityResultContracts.StartActivityForResult() to kontrakt który mówi, że oczekujemy wyniku z uruchomionej aktywności.
2. Drugi argument to ActivityResultCallback, który definiuje, co ma się stać po otrzymaniu wyniku. Tworzymy go jako klasę anonimową zawierającą metodę:

   onActivityResult(ActivityResult o) - jest ona wywoływana, gdy otwierana aktywność zakończy działanie i zwróci wynik.

```
activityLauncher = registerForActivityResult(
      new ActivityResultContracts.StartActivityForResult(),
      new ActivityResultCallback<ActivityResult>() {
        @Override
        public void onActivityResult(ActivityResult o) {
          //reakcja na odpowiedź aktywności
        }
      });
```

Uwaga na parametr typu ActivityResult. Będzie on wykorzystywany w obsłudze odpowiedzi, a przyjmuje różne nazwy w zależności od wersji Android studio na której pracujemy

Otwieranie aktywności i odbieranie od niej odpowiedzi

3. Kolejnym krokiem jest przygotowanie intencji która otworzy aktywność oraz wysłanie jej do sytemu.

Kontekst, który informuje kto wysłał intencję.

Czasami stosujemy sam wskaźnik this, ale ten sposób nie dział, jeżeli otwieramy aktywność bezpośrednio z wewnątrz słuchacza zdarzeń (wtedy this pokazuje na słuchacza nie na aktywność)

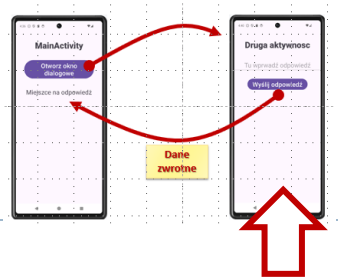
Można też zastosować funkcję getAplicationContext()

```
Intent intencja = new Intent(MainActivity.this, DrugaAktywnosc.class);
activityLauncher.launch(intencja);
```

Wysłanie intencji do systemu

Klasa którą otwieramy

Otwieranie aktywności i odbieranie od niej odpowiedzi



Przygotowanie odpowiedzi – w aktywności podrzędnej

```
Intent odpowiedz = new Intent();
odpowiedz.putExtra("result", "Treść odpowiedzi");
setResult(RESULT_OK, odpowiedz);
finish();
```

Odpowiedzią zwracaną przez okno jest intencja. Tworzymy intencję bez żadnych parametrów, gdyż nie otwiera ona nowej aktywności.

Intencja ta pełni dwie role:

1. informuje system, że należy przywrócić aktywność nadrzędną,
2. przenosi do aktywności nadrzędnej dane, które do niej do dołączymy.

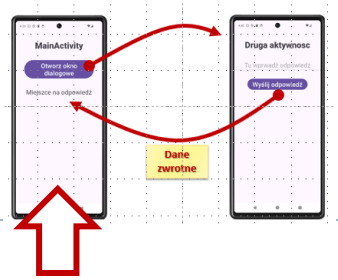
Odpowiedź wysyłamy do systemu za pomocą funkcji setResult().

Pierwszym jej parametrem jest kod rezultatu (*RESULT_OK* lub *RESULT_CANCELED*)

Po wysłaniu odpowiedzi zwykle zamykamy aktywność - finish();

Do intencji można dołączyć dane dowolnego typu – np. Informacje wpisane w oknach dialogowych

Otwieranie aktywności i odbieranie od niej odpowiedzi



Odbieranie odpowiedzi – w aktywności startowej

```
TextView textView = findViewById(R.id.textView01);

activityLauncher = registerForActivityResult(
        new ActivityResultContracts.StartActivityForResult(),
        new ActivityResultCallback<ActivityResult>() {
            @Override
            public void onActivityResult(ActivityResult o) {
                if (o.getResultCode() == RESULT_OK && o.getData() != null) {
                    String trescOdpowiedzi = o.getData().getStringExtra("result");
                    textView.setText(trescOdpowiedzi);
                }
            }
        });
```

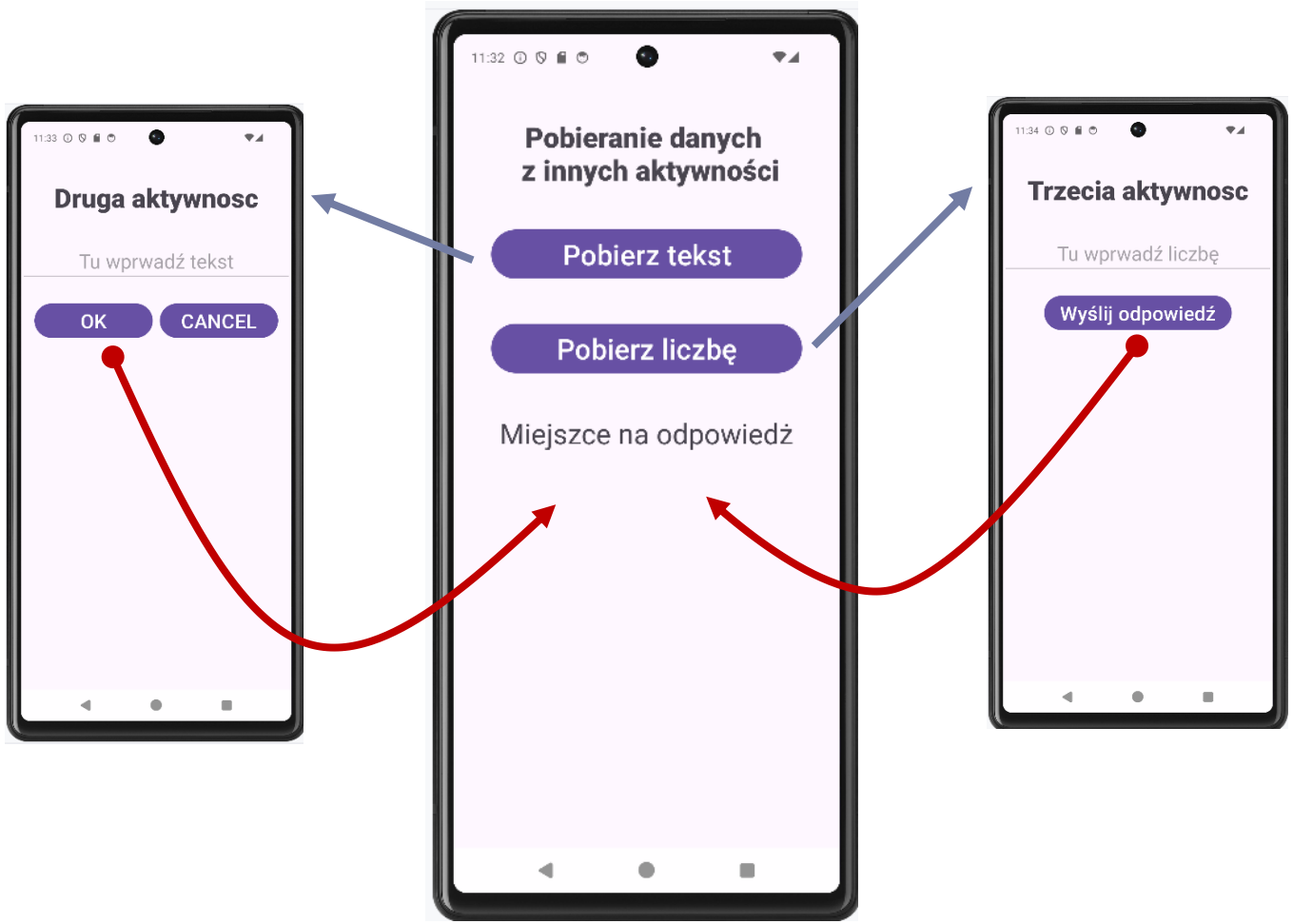
Po zamknięciu okna dialogowego uruchomiona jest metoda onActivityResult(). Przyjmuje ona w parametrze obiekt o nazwie „o”

Przed odebraniem odpowiedzi należy sprawdzić kot odpowiedzi oraz na wszelki wypadek czy dołączone są do niej jakieś dane

Z obiektu odpowiedzi możemy wyciągnąć doczepiony do nich dane.

Należy znać typy danych oraz klucze które, pod którymi je dodaliśmy.

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD



Przykład aplikacji posiadającej 2 aktywności podrzędne jedna z nich służy do pobierania tekstu druga pobiera liczbę

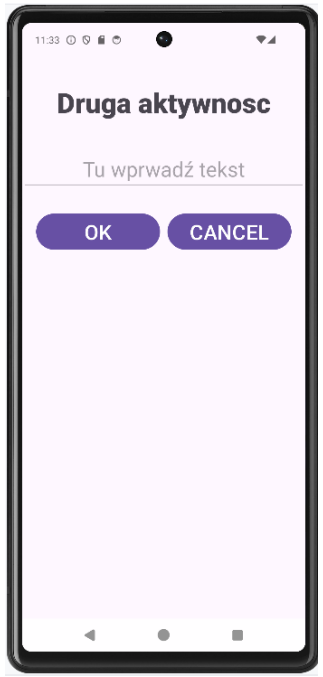
Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD


```
<TextView
   android:id="@+id/textView00"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Pobieranie danych \n z innych aktywności"
   app:layout_constraintLeft_toLeftOf="parent"
   app:layout_constraintRight_toRightOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   android:layout_marginTop="40dp"
   android:textSize="30sp"
   android:fontFamily="sans-serif-black"
   android:gravity="center"/>

<Button
   android:id="@+id/button01"
   android:text="Pobierz tekst"
   android:layout_width="0dp"
   android:layout_height="wrap_content"
   app:layout_constraintEnd_toEndOf="parent"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toBottomOf="@id/textView00"
   android:layout_margin="40dp"
   android:textSize="30sp"
   />
```

```
<Button
   android:id="@+id/button02"
   android:text="Pobierz liczbę"
   android:layout_width="0dp"
   android:layout_height="wrap_content"
   app:layout_constraintEnd_toEndOf="parent"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toBottomOf="@id/button01"
   android:layout_margin="40dp"
   android:textSize="30sp"
   />
<TextView
   android:id="@+id/textView01"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Miejsce na odpowiedź"
   app:layout_constraintLeft_toLeftOf="parent"
   app:layout_constraintRight_toRightOf="parent"
   app:layout_constraintTop_toBottomOf="@id/button02"
   android:layout_marginTop="40dp"
   android:textSize="30sp"/>
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD




```
<com.google.android.material.textfield.TextInputEditText
    android:id="@+id/EditText01_activity_druga"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/textView00_activity_druga"
    android:layout_marginTop="40dp"
    android:hint="Tu wprwadź tekst"
    android:textSize="30sp"
    android:gravity="center"
    />
```

```
<Button
    android:id="@+id/button01_activity_druga"
    android:text="OK"
    android:layout_width="180dp"
    android:layout_height="wrap_content"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/EditText01_activity_druga"
    android:layout_marginTop="30dp"
    android:layout_marginLeft="20dp"
    android:textSize="30sp"/>

<Button
    android:id="@+id/button02_activity_druga"
    android:text="CANCEL"
    android:layout_width="180dp"
    android:layout_height="wrap_content"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@id/EditText01_activity_druga"
    android:layout_marginTop="30dp"
    android:layout_marginRight="20dp"
    android:textSize="30sp"/>
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD


```
<TextView
   android:id="@+id/textView00_activity_trzecia"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Trzecia aktywnosc"
   app:layout_constraintLeft_toLeftOf="parent"
   app:layout_constraintRight_toRightOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   android:layout_marginTop="40dp"
   android:textSize="40sp"
   android:fontFamily="sans-serif-black"
   android:gravity="center"
   />

<com.google.android.material.textfield.TextInputEditText
   android:id="@+id/EditText01_activity_trzecia"
   android:layout_width="0dp"
   android:layout_height="wrap_content"
   app:layout_constraintEnd_toEndOf="parent"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toBottomOf="@id/textView00_activity_trzecia"
   android:layout_marginTop="40dp"
   android:hint="Tu wprwadź liczbę"
   android:textSize="30sp"
   android:gravity="center"
   android:inputType="number"
   />
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD


```
public class MainActivity extends AppCompatActivity {
//Przygotowujemy referencje dla 2 launcherów, każdy z nich będzie obsługiwał swoją aktywność
   private ActivityResultLauncher<Intent> activityLauncher_01;
   private ActivityResultLauncher<Intent> activityLauncher_02;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
     //....
      TextView textView = findViewById(R.id.textView01);
      Button button01 = findViewById(R.id.button01);
      Button button02 = findViewById(R.id.button02);
      //Tworzymy pierwszy luncher
      activityLauncher_01 = registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            new ActivityResultCallback<ActivityResult>() {
               @Override
               public void onActivityResult(ActivityResult o) {
                  //Kod, który wykonywany będzie gdy aktywność powiązana z pierwszym launcherem zwróci odpowiedź
                  if (o.getResultCode() == RESULT_OK && o.getData() != null) {
                     //Oczekujemy, że aktywność zwróci win cuch tekstu zapisany pod kluczem "resoult"
                     String resultData = o.getData().getStringExtra("result");
                     textView.setText(resultData);
                  }
               }
            });
```

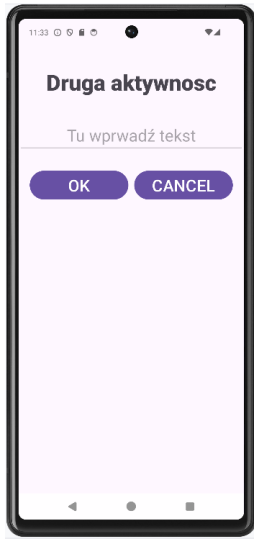
Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD


```
//Tworzymy drugi luncher
   activityLauncher_02 = registerForActivityResult(
         new ActivityResultContracts.StartActivityForResult(),
         new ActivityResultCallback<ActivityResult>() {
            @Override
            public void onActivityResult(ActivityResult o) {
               //Kod, który wykonywany będzie gdy aktywność powiązana z drugim
               // launcherem zwróci odpowiedź
               if (o.getResultCode() == RESULT_OK && o.getData() != null) {
                  //Oczekujemy danych typu integer
                  int resultData = o.getData().getIntExtra("result", 0);
                  textView.setText("Przesłana liczba = " + resultData);
               }
            }
         });
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD


```
//Słuchacz zdarzeń dla obu przycisków otwierających aktywności
    View.OnClickListener sluchacz = new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        //To, który przycisk naciśnięto rozpoznajemy na podstawie jego id
        int id=v.getId();
        if(id==R.id.button01)
        {
          //Tworzymy intencję która uruchomi drugą aktywność
          Intent intent = new Intent(MainActivity.this, DrugaAktywnosc.class);
          //Rejestrujemy Intencją do pierwszego launchera jednocześnie wysyłając ją do
          // systemu, co powoduje otwarcie aktywności
          activityLauncher_01.launch(intent);
        }
        else if(id==R.id.button02)
        {
          //Analogicznie dla trzeciej aktywności i odpowiadającego jej lunchera
          Intent intent = new Intent(MainActivity.this, TrzeciaAktywnosc.class);
          activityLauncher_02.launch(intent);
        }
      }
    };
    button01.setOnClickListener(sluchacz);
    button02.setOnClickListener(sluchacz);
  }
}
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD



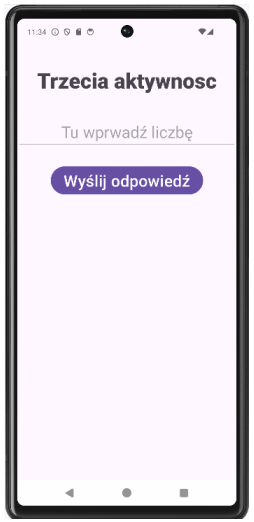

```
// Aktywność zwraca odpowiedź w postaci łańcucha znaków posiada
// dwa przyciski: OK i Cancel
// Przycisk OK zatwierdza wysłanie odpowiedzi (łańcuch może być pusty)
// Przycisk Cancel zamyka aktywność



public class DrugaAktywnosc extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_druga_aktywnosc);
// ...
        EditText editText = findViewById(R.id.EditText01_activity_druga);
        Button button_OK = findViewById(R.id.button01_activity_druga);
        Button button_CANCEL = findViewById(R.id.button02_activity_druga);
```

```
// Tworzymy słuchacza dla przycisków OK i Cancel
    View.OnClickListener sluchacz = new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            int id = v.getId();
            if (id == R.id.button01_activity_druga) //Kliknięto przycisk OK
            {
                //Pobieramy odpowiedź z pola tekstowego
                String text = editText.getText().toString();
                //Przygotowujemy intencję która Przeniesie odpowiedź
                Intent odpowiedz = new Intent();
                //Dodajemy pobrany tekst do intencji
                odpowiedz.putExtra("result", text);
                //Wysyłamy intencję z kodem RESULT_OK
                setResult(RESULT_OK, odpowiedz);
            }
            else if (id == R.id.button02_activity_druga) //Kliknięto przycisk Cancel
            {
                //Wysyłamy kod RESOULT_CANCELED
                setResult(RESULT_CANCELED);
            }
            //Zamykamy aktywność
            finish();
        }
    };
    button_OK.setOnClickListener(sluchacz);
    button_CANCEL.setOnClickListener(sluchacz);
}
}
```

Otwieranie aktywności i odbieranie od niej odpowiedzi - PRZYKŁAD

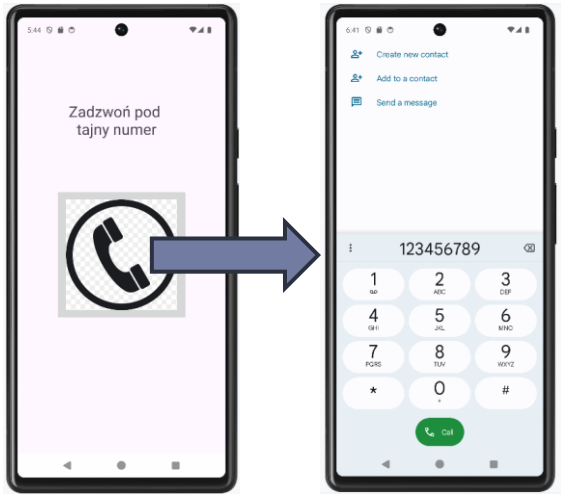



```
// Aktywność zwraca odpowiedź w postaci liczby całkowitej
// Jeżeli pole dialogowe będzie puste użytkownik otrzyma informację
// o błędzie, a aktywność nie zostanie zamknięta

public class TrzeciaAktywnosc extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_trzecia_aktywnosc);
// ...
        EditText editText = findViewById(R.id.EditText01_activity_trzecia);
        Button button = findViewById(R.id.button02_activity_trzecia);

//Tworzymy słuchacza dla przycisku
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Sprawdzamy czy pole tekstowe nie jest puste
                if(!editText.getText().toString().isEmpty())
                {
                    //Pobieramy odpowiedź z pola tekstowego
                    int liczba = Integer.parseInt(editText.getText().toString());
                    //Przygotowujemy intencję, która przeniesie odpowiedź
                    Intent odpowiedz = new Intent();
                    //Dodajemy pobraną liczbę do intencji
                    odpowiedz.putExtra("result", liczba);
                    //Wysyłamy intencję z kodem RESULT_OK
                    setResult(RESULT_OK, odpowiedz);
                    //Zamykamy aktywność
                    finish();
                }
                else
                {
                    //w przeciwnym razie wyświetlamy komunikat o błędzie
                    Toast.makeText(TrzeciaAktywnosc.this, "Musisz podać odpowiedź",
                            Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Otwieranie aplikacji telefon



Tworzymy intencję z akcją ACTION_DIAL, która informuje system, że chcemy otworzyć aplikację telefonu w trybie wybierania numeru.

```
ImageButton button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String numerTelefonu = "123456789";
        Intent intent = new Intent(Intent.ACTION_DIAL);
        intent.setData(Uri.parse("tel:" + numerTelefonu));
        startActivity(intent);
    }
});
```

Ustawiamy dane intencji na URI z prefiksem tel: i numerem telefonu.

Uruchamiamy aktywność z utworzoną intencją.

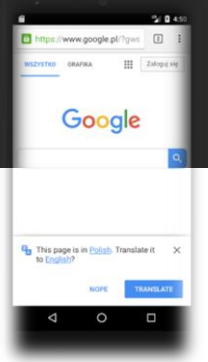
Jeśli chcesz automatycznie rozpocząć połączenie, użyj akcji ACTION_CALL. Jednak wymaga to dodania uprawnienia CALL_PHONE do pliku AndroidManifest.xml.

Przykład: niejawne otwieranie aktywności - URL

W poprzednich przykładach wskazywaliśmy, którą aktywność chcemy uruchomić. Możemy jednak także zdefiniować zadanie do wykonania, a wybór Aktywności, która je obsłuży pozostawić systemowi (i nie musi być to aktywność nasze aplikacji)

Do Intencji przekażemy jedynie akcję oraz dane, na których będziemy chcieli ją przeprowadzić.

```
91      private void otworzURI() {
92          Uri adres = Uri.parse("http://www.google.pl");
93          Intent intencja = new Intent(Intent.ACTION_VIEW, adres);
94          //intencja.setData(adres);
95          startActivity(intencja);
96      }
```



- Akcją którą chcemy wykonać jest Intent.ACTION_VIEW czyli obejrzyj.
- Danymi, na których wykonamy akcję jest adres url (ogólniej adres URI)

Takie wywołanie nie wymaga tworzenia nowej aktywności – przeglądarka jest w standardzie

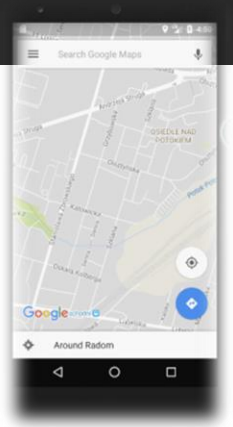
Przykład: niejawne otwieranie aktywności – współrzędne geograficzne

Innym typem danych URI, które mogą być obsłużone przez system jest geoURI – czyli współrzędne geograficzne

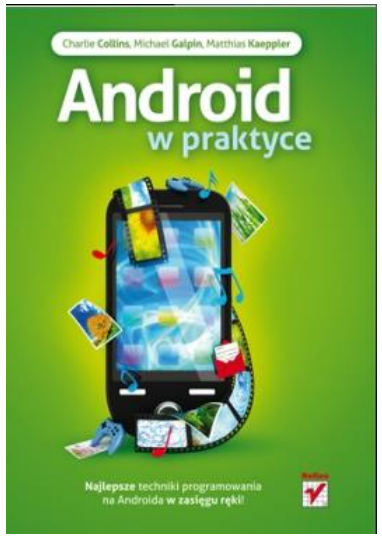
```
85      private void otworzGeoURI() {
86          Uri geoAdres = Uri.parse("geo:51.405,21.1756");
87          Intent intencja = new Intent(Intent.ACTION_VIEW,geoAdres);
88          startActivity(intencja);
89      }
```

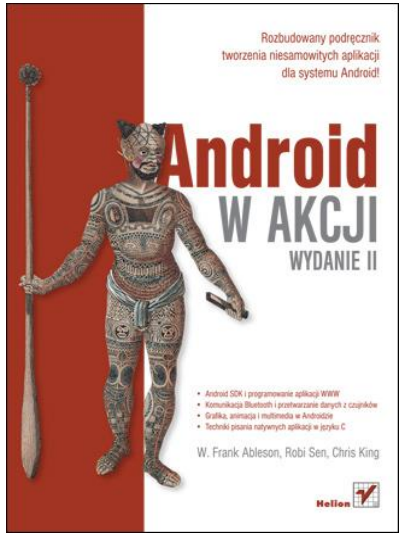
Współrzędne należy przekonwertować na standardowe zmienną typu Uri za pomocą metody Uri.parse( )

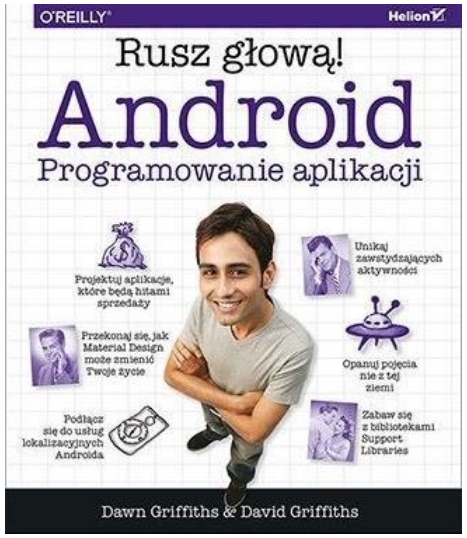
Takie wywołanie również nie wymaga tworzenia nowej aktywności – mapa Google też jest w standardzie

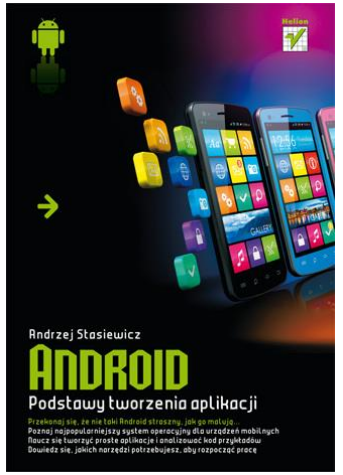


Literatura


Charlie Collins, Michael Galpin, Matthias Kaeppler
Android
w praktyce
Najlepsze techniki programowania
na Androida w zasięgu ręki!


Rozbudowany podręcznik
tworzenia niesamowitych aplikacji
dla systemu Android!
Android
W AKCJI
WYDANIE II
W. Frank Ableson, Robi Sen, Chris King
Helion


O'REILLY
Helion
Rusz głową!
Android
Programowanie aplikacji
Projektuj aplikacje, które będą hitami sprzedaży
Unikaj zawstydzających aktywności
Przekonaj się, jak Material Design może zmienić Twoje życie
Opanuj pojęcia nie z tej ziemi
Podłącz się do usług lokalizacyjnych Androida
Zabaw się z bibliotekami Support Libraries
Dawn Griffiths & David Griffiths


Helion
Andrzej Stasiewicz
ANDROID
Podstawy tworzenia aplikacji