Uniwersytet Wrocławski

Rafał Lewandków

PROGRAMOWANIE URZĄDZEŃ MOBILNYCH

WYKŁAD 1

Fundamenty Aplikacji 1

- Context
- Aktywność
- Cykl Życia
- Mechanizm Komunikacji Międzyprocesowej - Intent
- Layout

Podstawowe elementy aplikacji

Podstawowe elementy – komponenty, klasy pierwotne z których zbudowana jest aplikacja (VM + Zasoby = APK (Android Package Kit))

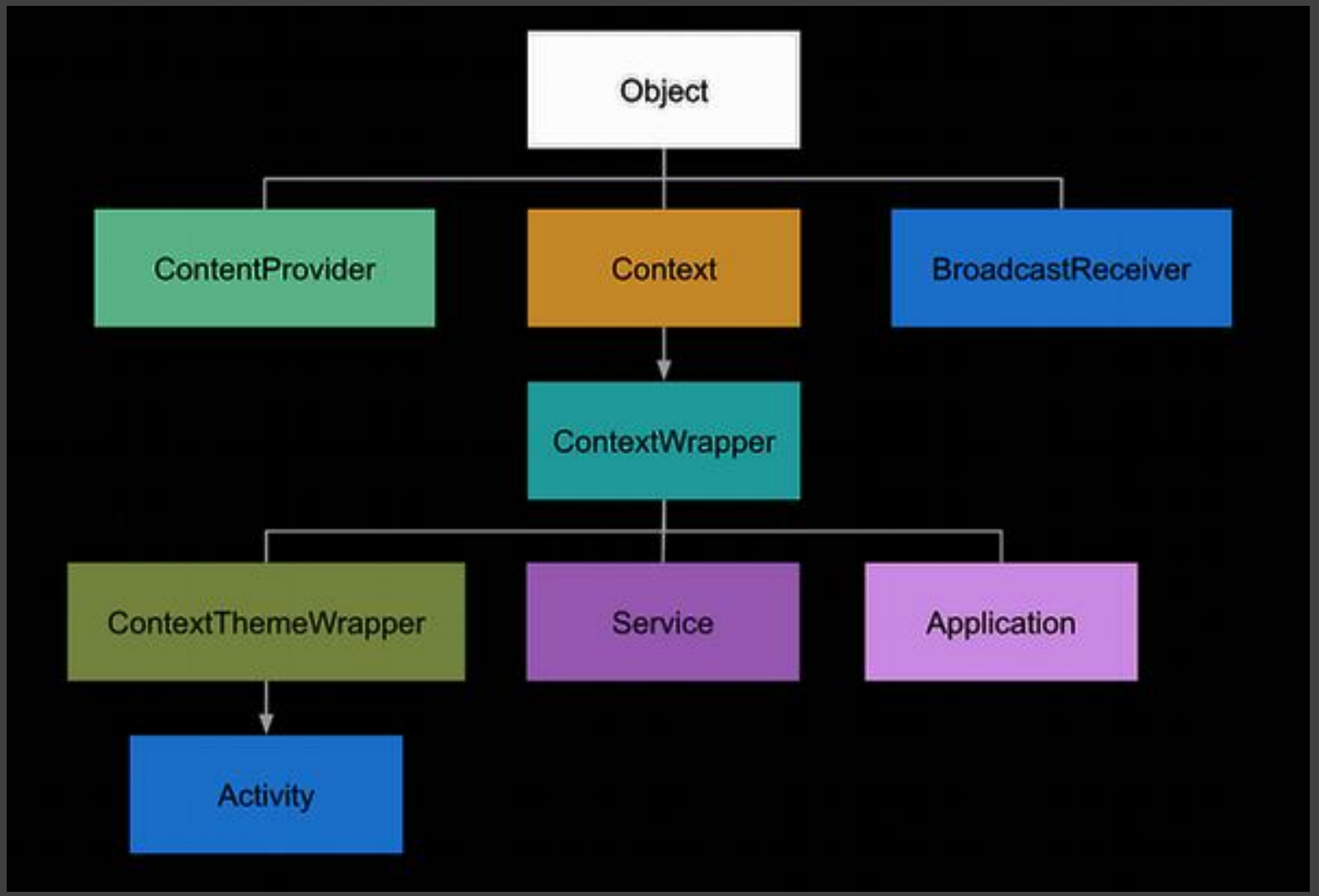
Każdy komponent zapewnia konkretną funkcjonalność i posiada wyodrębniony cykl życia. Komponenty działają kooperacyjnie, wspólnie zapewniając ukończenie określonego zadania aplikacji

Podstawowe elementy:

1. Activities/Fragments – pojedynczy ekran z UI
2. Services – Usługi są często wykorzystywane do obsługi długotrwałych operacji, zarządzania komunikacją między komponentami aplikacji oraz wykonywania operacji, które mają trwać po zamknięciu interfejsu użytkownika aplikacji
3. Broadcast Receiver – pozwala aplikacji na reagowanie na różnego rodzaju zdarzenia lub komunikaty systemowe
4. Content Provider – umożliwia aplikacjom dzielenie się danymi z innymi aplikacjami w sposób kontrolowany i zabezpieczony

Uniwersytet Wrocławski

Podstawowe elementy aplikacji - Context



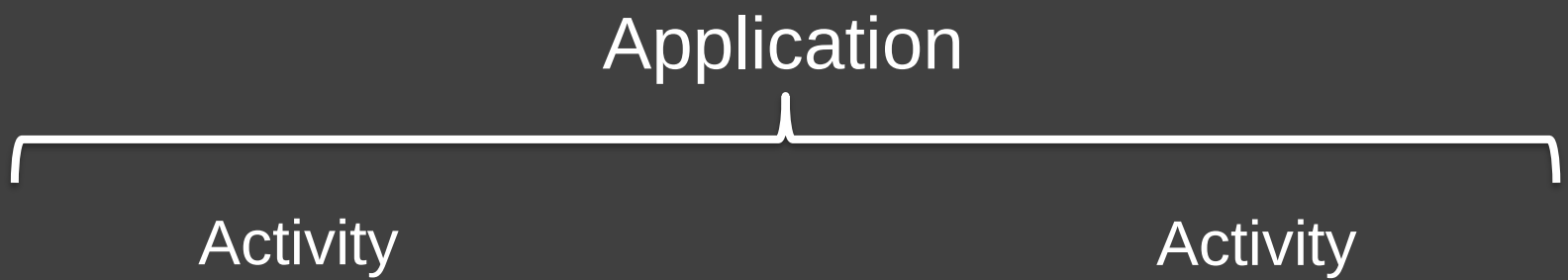
Uniwersytet Wrocławski

Podstawowe elementy aplikacji - Context

Application

- Globalny dostęp: Application Context dostarcza globalny dostęp do zasobów aplikacji np. takich jak pliki zasobów (katalog *res,* klasa R)
- Długi cykl życia: Application Context jest tworzony raz podczas uruchamiania aplikacji i pozostaje dostępny przez cały czas życia aplikacji. Nie jest on powiązany z cyklem życia konkretnego Activity, więc można go używać nawet po zamknięciu Activity.
- Bezpieczne do użycia w dłuższych operacjach: Application Context jest bezpieczny do użycia w dłuższych operacjach, które mogą być wykonywane w tle, ponieważ nie jest związany z cyklem życia UI i nie powinien prowadzić do wycieków pamięci.

Podstawowe elementy aplikacji - Context

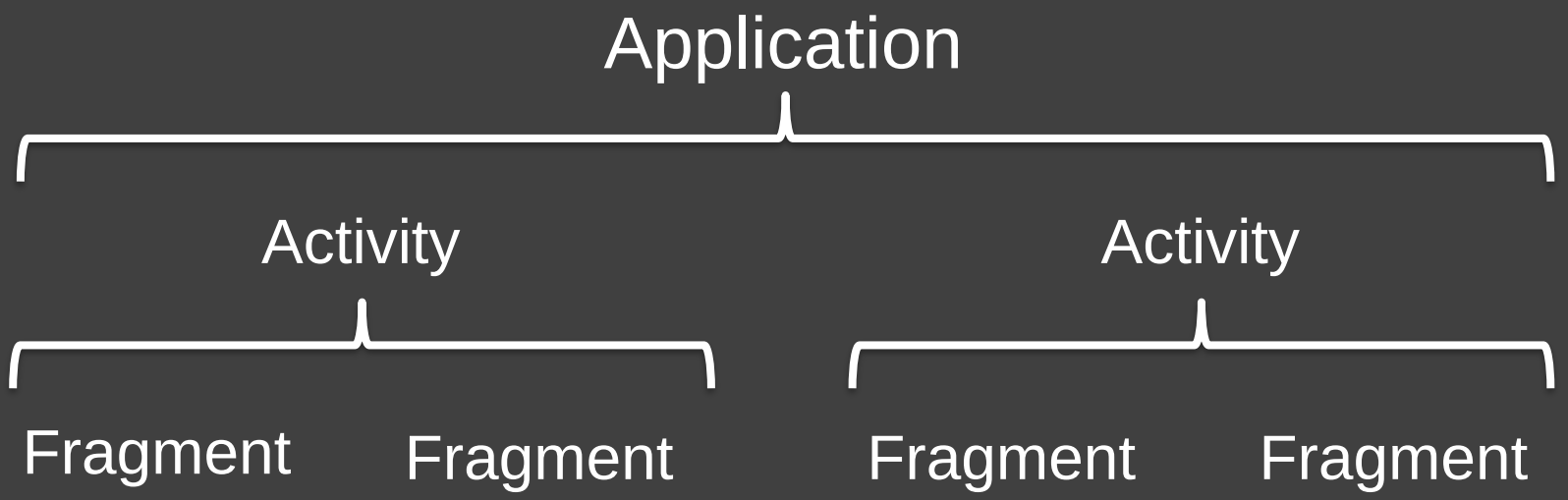


1. **Związek z aktywnością**: Kontekst aktywności jest bezpośrednio związany z określoną aktywnością i działa w kontekście tej aktywności. Oznacza to, że jest dostępny tylko wtedy, gdy dana aktywność jest aktywna.

2. **Ograniczony do cyklu życia aktywności**: Kontekst aktywności ma ograniczony okres życia, który jest związany z cyklem życia aktywności. Kiedy aktywność jest zniszczona, kontekst aktywności staje się nieważny.

3. **Dostęp do widoków**: Kontekst aktywności umożliwia dostęp do widoków (View) znajdujących się w obrębie danej aktywności. Dzięki temu można manipulować widokami, np. zmieniać ich właściwości czy obsługiwać zdarzenia.

Podstawowe elementy aplikacji - Context



1. **Ograniczony do fragmentu**: Kontekst fragmentu jest dostępny tylko w obrębie danego fragmentu i nie jest dostępny poza tym fragmentem. Oznacza to, że nie można go używać poza zakresem tego fragmentu.

2. **Dostęp do widoków**: Kontekst fragmentu umożliwia dostęp do widoków (View) znajdujących się wewnątrz tego fragmentu. Można używać go do manipulowania widokami, np. zmieniać ich właściwości czy obsługiwać zdarzenia.

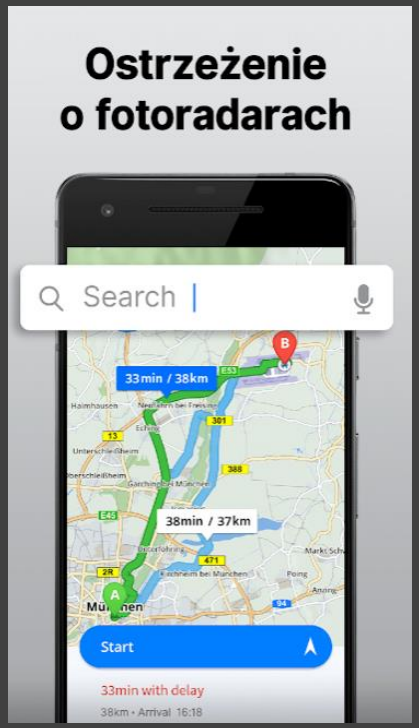
Podstawowe elementy aplikacji

- Activity Class – obiekt, które poza wyświetlaniem / zbieraniem danych zapewnia pewną funkcjonalność
- Typowo aplikacje zawierają jeden obiekt *Activity* i wiele Fragmentów
- Aplikacje muszą wyznaczyć jedną Aktywność jako *main task/entry point.* Jest uruchamiana jako pierwsza gdy aplikacja jest włączana
- Aktywność może przekazać kontrolę i dane do innej Aktywności poprzez protokół komunikacji międzyprocesowej *intents*

BACHA Soft

Podstawowe elementy aplikacji

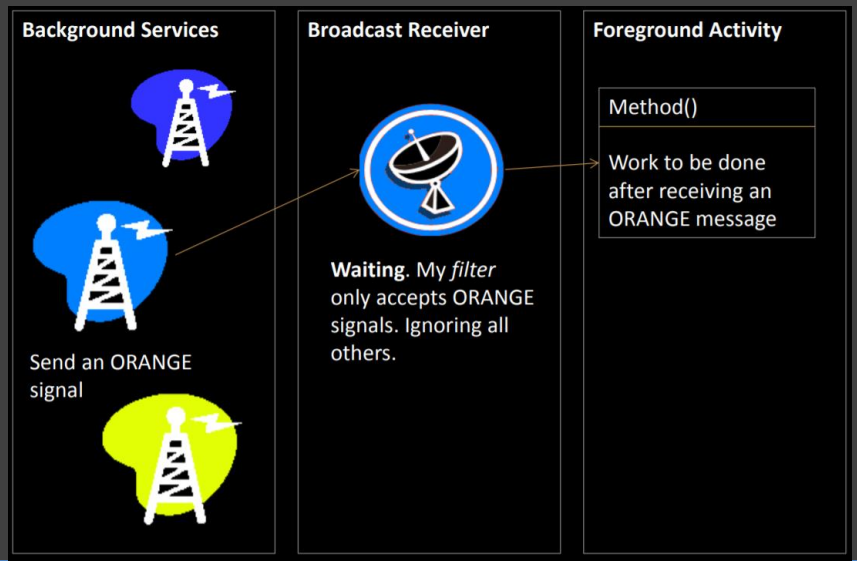
- Service Class – specjalny typ aktywności – nie posiada *Visual User Interface*. Usługa może być aktywna w tle
- Usługi zazwyczaj pracują w tle wykonując „busy-work” przez nieokreślony /nieskończony czas
- Aplikacje rozpoczynają własne usługi lub łączą się z usługami już aktywnymi (GPS, przełączanie między aplikacjami, aktywnościami, brak okienek)



Maps, GPS Navigation

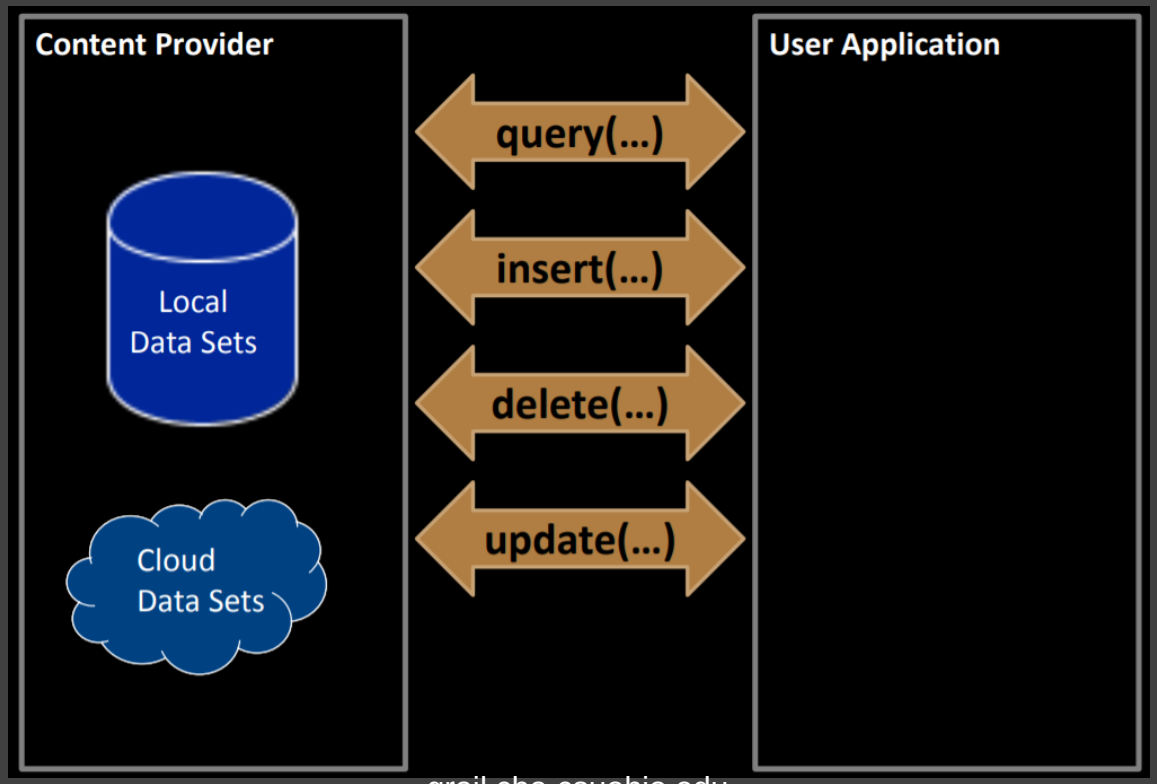
Podstawowe elementy aplikacji

- BroadcasterReceiver Class – dedykowany *listener* oczekujący na wiadomość zwykle typu *system-wide* w celu wykonania pewnych czynności
- Przykładowo: niski poziom baterii, wi-fi dostępne
- Nie posiadają interfejsu
- Rejestrowanie wiadomości, typowo po kluczu – jeśli wiadomość odpowiada kluczowi odbiornik jest aktywowany
- Zazwyczaj odpowiada poprzez wykonanie specjalnej czynności lub przekazanie użytkownikowi wiadomości/powiadomienia



Podstawowe elementy aplikacji

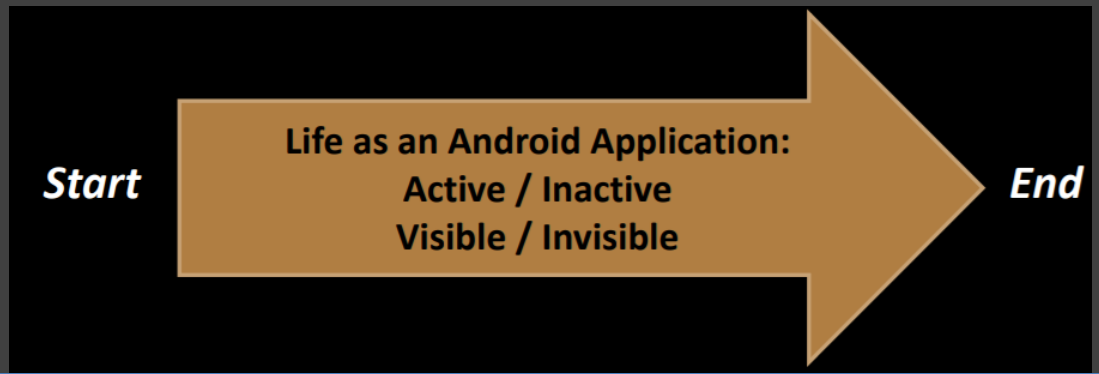
- ContentProvider Class – zapewnia dostęp do danych wielu aplikacjom
- Dane globalne: lista kontaktów, zdjęcia, wiadomości, filmy, email
- Dane globalne są zazwyczaj przechowywane w bazie danych (SQLite)
- Klasa oferuje metody innym aplikacjom: retrieve, delete, update, insert
- Wrapper ukrywający właściwe dane. Dostępny interfejs.



Cykl Życia

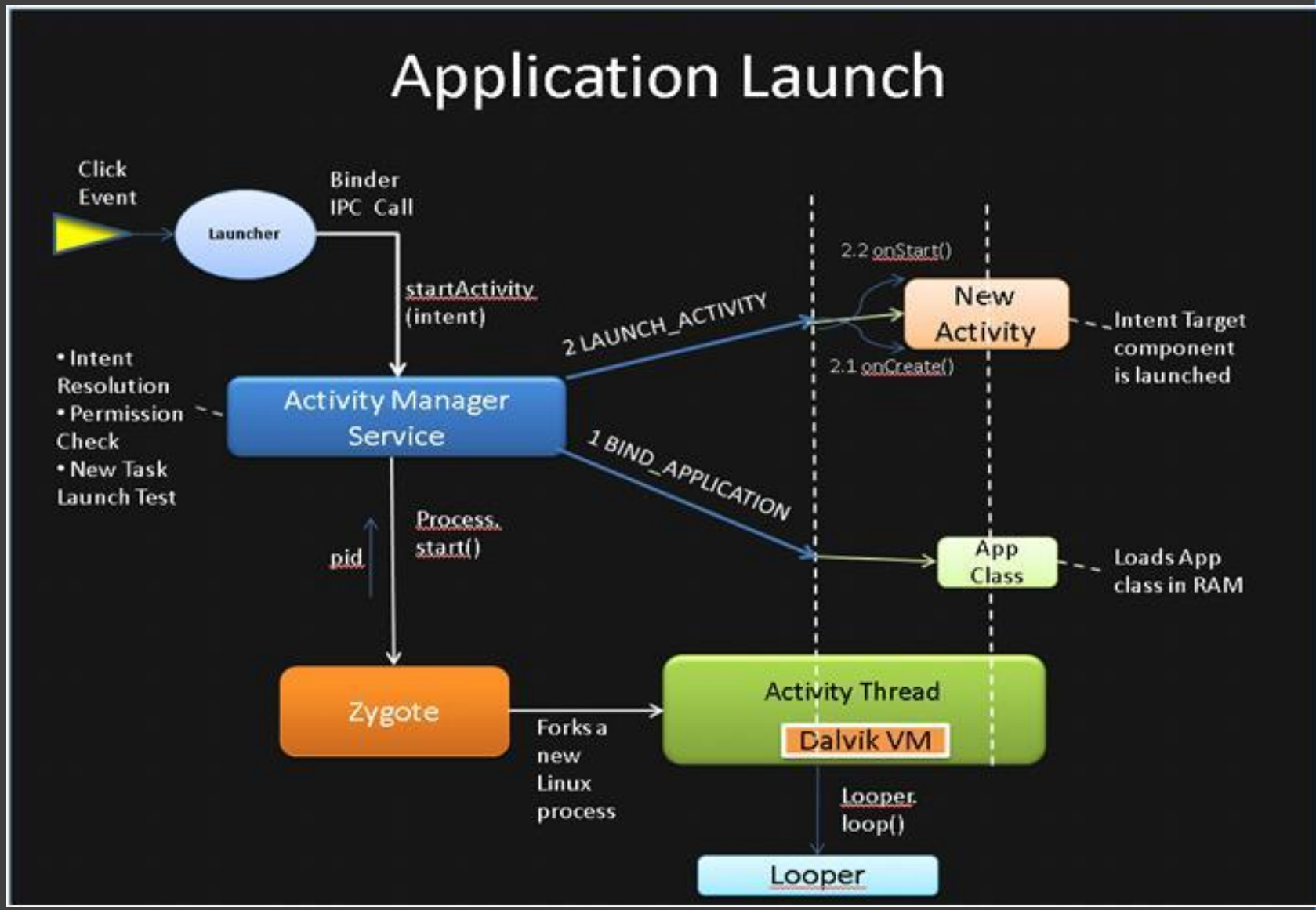
- Każda aplikacja Androida działa w swojej własnej instancji maszyny wirtualnej
- W każdym momencie kilka instancji maszyny wirtualnej może być aktywna (rzeczywista równoległość – nie *task switching*)
- Aplikacja Androida nie kontroluje całkowicie realizacji swojego cyklu życia

- OS może zakończyć każdy proces
  - Zasoby są krytycznie niskie
  - Duża liczba działających aplikacji
  - Aplikacja wymagająca bardzo dużych zasobów (energia, pamięć)



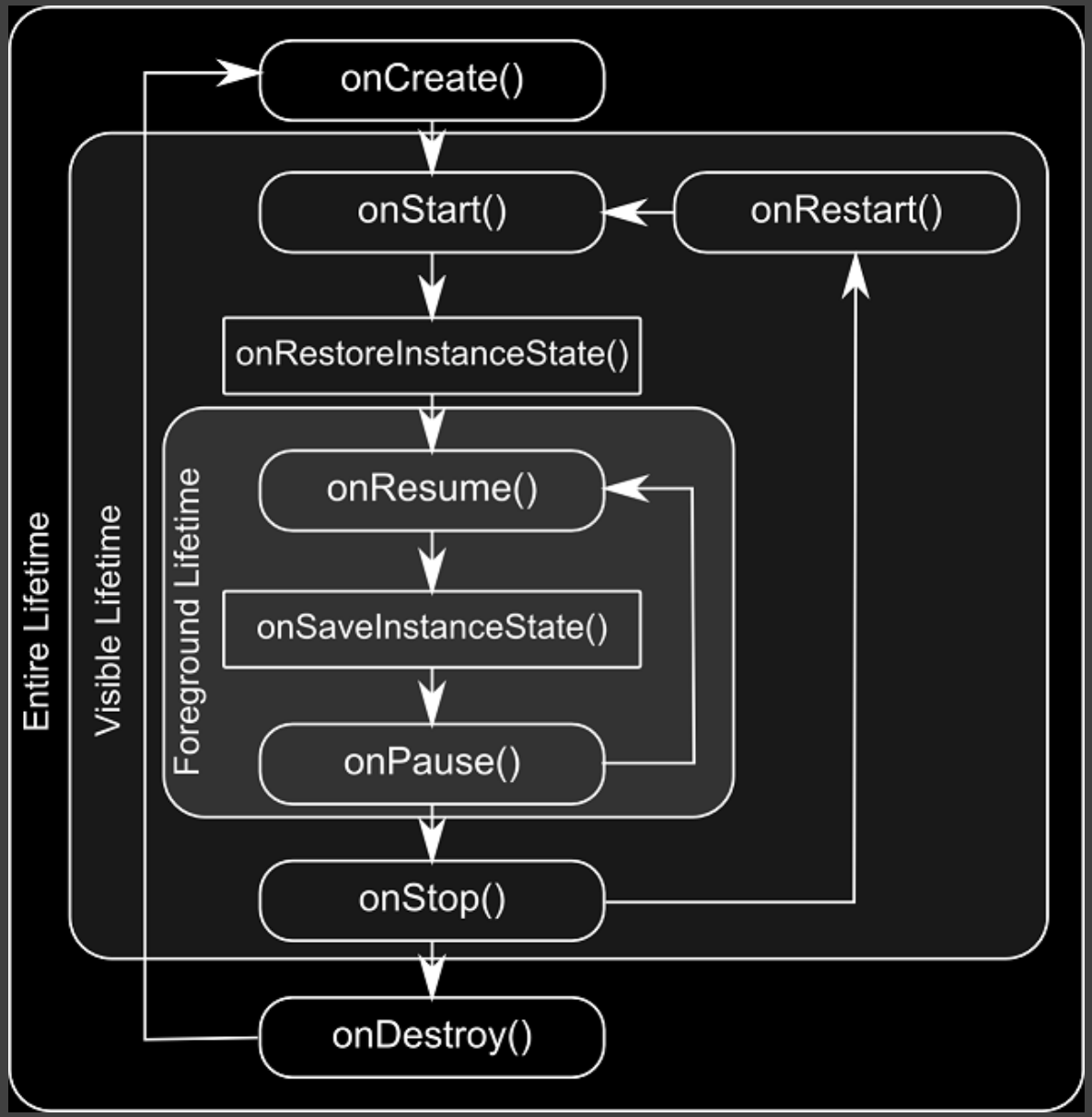
Uniwersytet Wrocławski

Cykl Życia



Uniwersytet Wrocławski

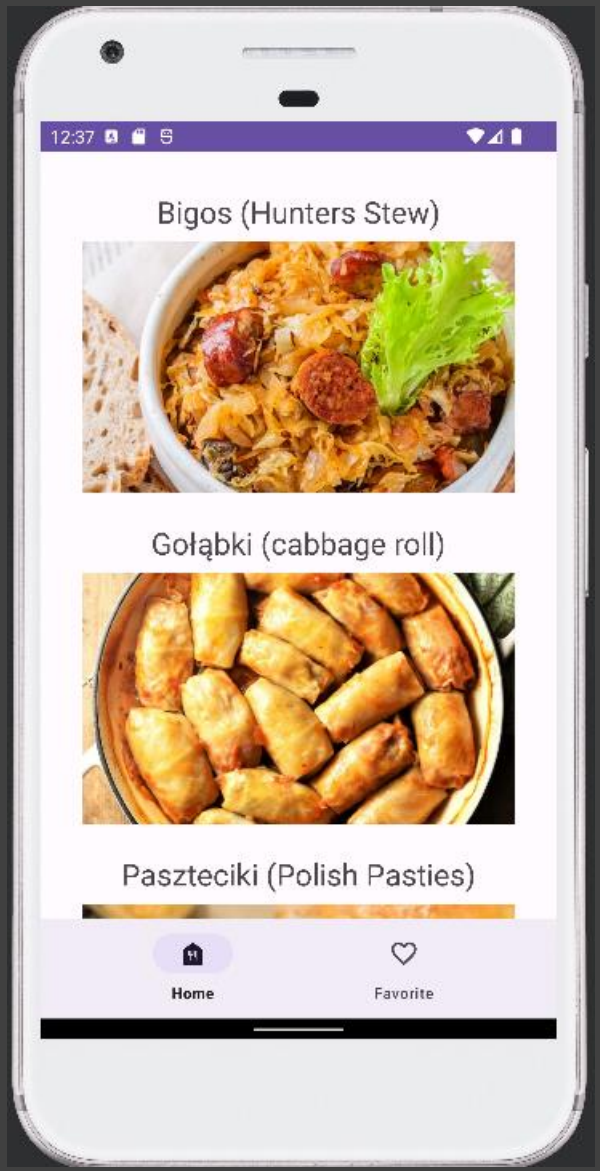
Cykl Życia



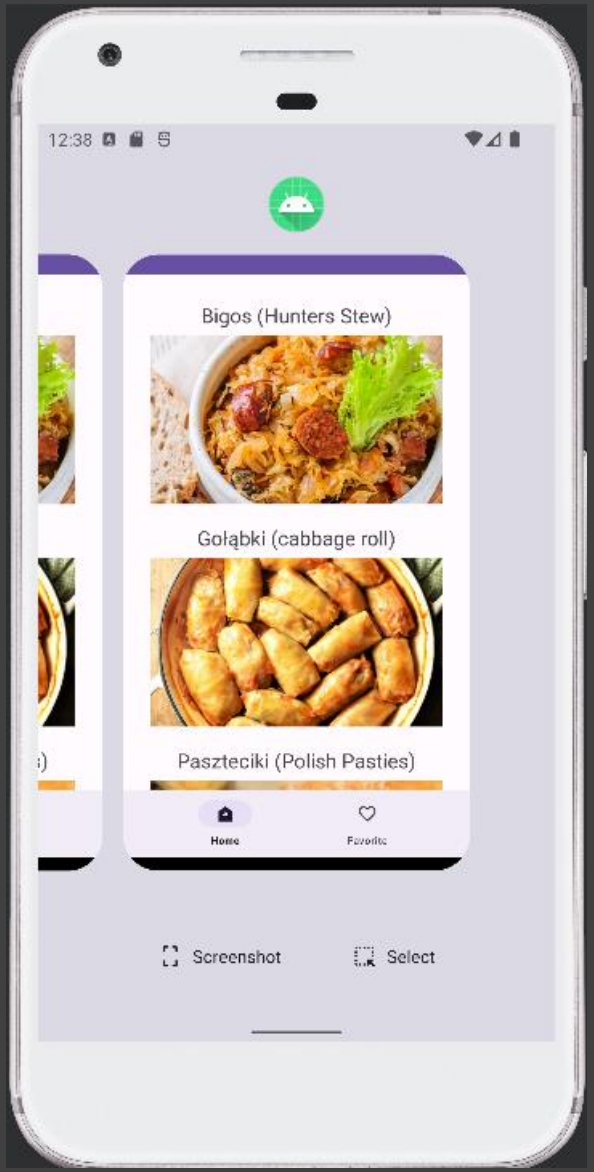
Uniwersytet Wrocławski

Cykl Życia

Foreground

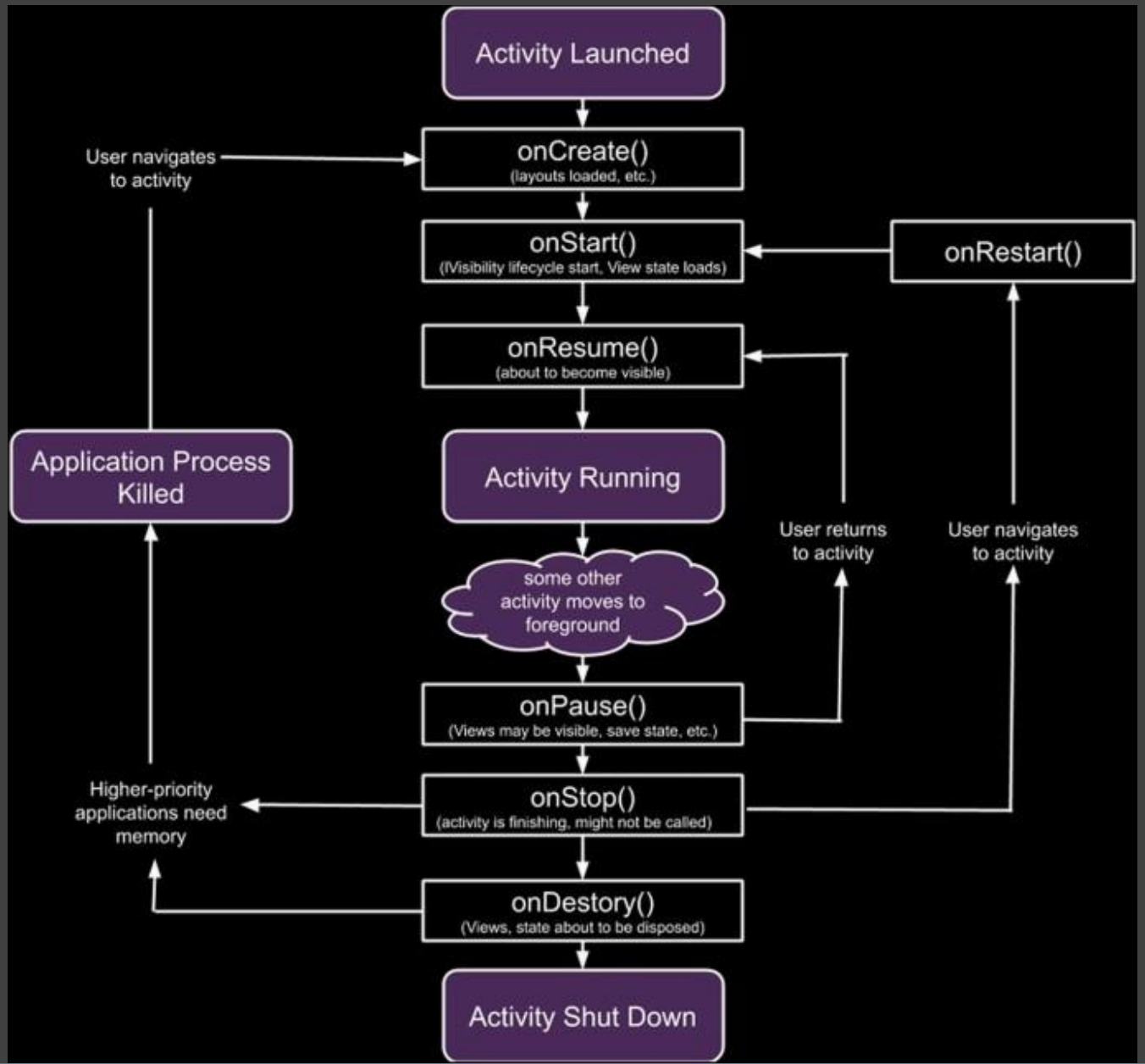


Visible



Uniwersytet Wrocławski

Cykl Życia



Uniwersytet Wrocławski

Cykl Życia

Zmiana konfiguracji wymaga przeładowania layoutu oraz innych zasobów gdy:

- Następuje zmiana orientacji urządzenia

Przy zmianie konfiguracji Android:

1. Wyłącza aktywność – onPause(), onStop(), onDestroy()
2. Startuje nową instancję aktywności – onCreate(), onStart(), onResume()

https://developer.android.com/reference/android/app/Activity.html

Uniwersytet Wrocławski

Cykl Życia – Trwałość Danych

Bundle Class

- Można zapisać nawet cały obiekt
- Klasa musi implementować interfejs `Parcelable`
  - Konstruktor przyjmuje obiekt typu `Parcel`
  - `writeToParcel(Parcel dest, int flags)`
  - `Parcelable.Creator`

https://developer.android.com/reference/android/app/Activity.html

Uniwersytet Wrocławski

Cykl Życia – Trwałość Danych

1.Przekazywanie danych między komponentami: Bundle jest często używany do przekazywania danych między różnymi komponentami aplikacji.

2.Zachowanie stanu: Bundle jest używany do zachowywania stanu komponentów w przypadku zmiany orientacji ekranu lub zniszczenia i odtworzenia, na przykład aktywności. Dzięki temu dane nie zostaną utracone podczas tych operacji.

3.Przekazywanie argumentów do fragmentów: Kiedy tworzysz fragmenty, możesz przekazywać im argumenty za pomocą obiektów Bundle.

4.Przekazywanie danych do usług i Broadcast Receiverów: Możesz używać Bundle do przekazywania danych do usług (Service) i Broadcast Receiverów, co pozwala na przetwarzanie tych danych w tle lub w odpowiedzi na zdarzenia systemowe.

5.Przekazywanie danych między komponentami aplikacji i aplikacjami: Bundle może być wykorzystywany do przekazywania danych między różnymi komponentami tej samej aplikacji lub nawet między różnymi aplikacjami w systemie Android, jeśli masz odpowiednie uprawnienia.

https://developer.android.com

Uniwersytet Wrocławski

Cykl Życia – Trwałość Danych

```
public class MainActivity extends AppCompatActivity {
    private static final String KEY_TEXT = "text_key";
    private String savedSave;

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

        if (savedInstanceState != null) {
            savedText = savedInstanceState.getString(KEY_TEXT);
        }
    }

    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        String textToSave = "text_value"
        outState.putString(KEY_TEXT, textToSave);
    }
}
```

https://developer.android.com

Uniwersytet Wrocławski

Mechanizm Komunikacji Międzyprocesowej - Intent

1.Komunikacja między komponentami: Intenty pozwalają na komunikację między różnymi komponentami aplikacji, takimi jak aktywności (Activity), fragmenty (Fragment), usługi (Service), oraz Broadcast Receivery.

2.Wykonywanie akcji: Intenty opisują intencje do wykonania określonej akcji, na przykład uruchomienie konkretnej aktywności, wysłanie wiadomości SMS lub otwarcie określonej strony internetowej.

3.Przekazywanie danych: Intenty mogą przekazywać dane jako dodatkowe informacje w formie paczki danych (Bundle).

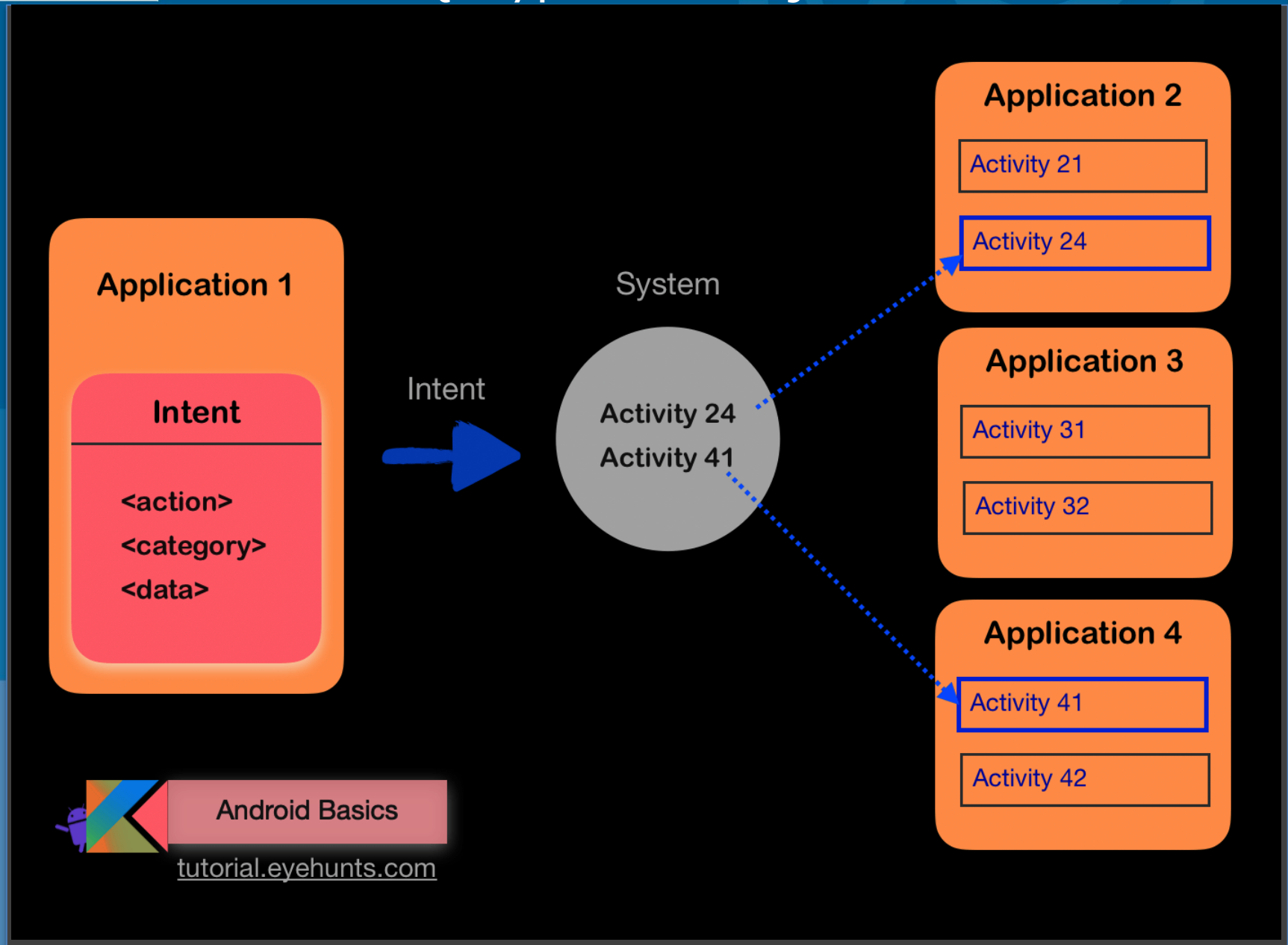
4.Rozgłaszanie zdarzeń (Broadcasting): Intenty mogą być używane do wysyłania wiadomości broadcast do innych komponentów systemu lub aplikacji. Pozwala to na powiadamianie innych komponentów o wystąpieniu określonych zdarzeń.

5.Startowanie komponentów: Intenty są używane do rozpoczęcia działania różnych komponentów, takich jak aktywności, usługi lub Broadcast Receiver.

6.Odpowiedzi (Result): Intenty mogą być używane do oczekiwania na odpowiedź (result) od innego komponentu.

Uniwersytet Wrocławski

Mechanizm Komunikacji Międzyprocesowej - Intent



Uniwersytet Wrocławski

Mechanizm Intencji

- Uruchamianie aktywności jest jednym z głównych zadań intencji
- Intencje uruchomieniowe zostały podzielone na dwa typy
  - Jawne – z jawnie określonym obiektem który chcemy otworzyć
    - Intencja uruchamiająca aktywność `mojaAktywność`

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

  - Domniemane – intencje zawierające informacje o tym co chcemy zrobić
    - Nie podajemy konkretnych klas które mają zadanie realizować
    - System decyduje która aktywność zostaje uruchomiona
    - Decyzje podejmuje przy pomocy Filtrów Intencji
    - Intencja pokazująca dane z adresu URL

```
Intent intent = new Intent (Intent.ACTION_VIEW,
                    Uri.parse(„www.uwr.edu.pl”));
```