Uniwersytet Wrocławski

Rafał Lewandków

PROGRAMOWANIE URZĄDZEŃ MOBILNYCH 1

WYKŁAD 8

- Fundamenty Aplikacji
- Aktywność
- Cykl Życia
- Jetpack Compose

Podstawowe komponenty

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

Podstawowe elementy:

- **Activities/Fragments/Composable** – pojedynczy ekran z UI
  - **Jetpack Compose** to **deklaratywna** biblioteka UI, która zastępuje klasyczne **XML + Views**, upraszczając budowę UI.

- **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

- **Broadcast Receiver** – pozwala aplikacji na reagowanie na różnego rodzaju zdarzenia lub komunikaty systemowe

- **Content Provider** – umożliwia aplikacjom dzielenie się danymi z innymi aplikacjami w sposób kontrolowany i zabezpieczony

Uniwersytet Wrocławski

Aktywność

W *„klasycznym”* Androidzie aktywność kontrolowała zarówno **logikę, jak i interfejs użytkownika (XML + Views).** W Jetpack Compose aktywność jest **tylko kontenerem**, który ustawia **Composable UI** i zarządza **cyklem życia aplikacji**. Aktywność to *„drzwi wejściowe”* do aplikacji – uruchamia pierwsze **Composable**.
Jetpack Compose **eliminuje** potrzebę **wielu aktywności** – zamiast nich nawigujemy między ekranami w **Composable**

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
```

Uniwersytet Wrocławski

Aktywność

Główna aktywność aplikacji, może hostować **Composable UI**

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Uniwersytet Wrocławski

Aktywność

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Uniwersytet Wrocławski

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Zapewnia poprawne działanie aktywności, wywołując **bazową implementację**.

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Uniwersytet Wrocławski

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Zapewnia poprawne działanie aktywności, wywołując **bazową implementację**.

Funkcja ułatwiająca wyświetlanie UI na **pełnym ekranie**, rozciągając zawartość na paski systemowe (status bar i nawigacja).

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Uniwersytet Wrocławski

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Zapewnia poprawne działanie aktywności, wywołując **bazową implementację**.

W jego wnętrzu umieszczamy **Composable UI**.

Funkcja ułatwiająca wyświetlanie UI na **pełnym ekranie**, rozciągając zawartość na paski systemowe (status bar i nawigacja).

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Uniwersytet Wrocławski

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Zapewnia poprawne działanie aktywności, wywołując **bazową implementację**.

W jego wnętrzu umieszczamy **Composable UI**.

Funkcja ułatwiająca wyświetlanie UI na **pełnym ekranie**, rozciągając zawartość na paski systemowe (status bar i nawigacja).

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Stosuje **generowany automatycznie** styl aplikacji, np. kolory, czcionki, motyw jasny/ciemny.

Uniwersytet Wrocławski

Aktywność

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Box(
                    modifier = Modifier.fillMaxSize(),
                    contentAlignment = Alignment.Center
                ) {
                    Text(text = "Hello World!")
                }
            }
        }
    }
}
```

Główna aktywność aplikacji, może hostować **Composable UI**

Metoda **cyklu życia** aktywności – jest wywoływana, gdy aktywność jest tworzona.

Zapewnia poprawne działanie aktywności, wywołując **bazową implementację**.

Funkcja ułatwiająca wyświetlanie UI na **pełnym ekranie**, rozciągając zawartość na paski systemowe (status bar i nawigacja).

W jego wnętrzu umieszczamy **Composable UI**.

Struktura danych używana do **przechowywania** i **przekazywania** informacji między **komponentami aplikacji**

Stosuje **generowany automatycznie** styl aplikacji, np. kolory, czcionki, motyw jasny/ciemny.

Kontener układający elementy **warstwowo**.

Podstawowy komponent do wyświetlania tekstu w **Jetpack Compose**.

AndroidManifest.xml

Kluczowy **plik konfiguracyjny** każdej aplikacji. Zawiera informacje niezbędne dla systemu Android, takie jak:

- Deklaracja aktywności, usług, odbiorników zdarzeń (Activity, Service, BroadcastReceiver)
- Uprawnienia wymagane przez aplikację (np. dostęp do Internetu, aparatu)

Deklaracja aktywności.

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:theme="@style/Theme.MyApplication">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

AndroidManifest.xml

Kluczowy **plik konfiguracyjny** każdej aplikacji. Zawiera informacje niezbędne dla systemu Android, takie jak:

- Deklaracja aktywności, usług, odbiorników zdarzeń (Activity, Service, BroadcastReceiver)
- Uprawnienia wymagane przez aplikację (np. dostęp do Internetu, aparatu)

Deklaracja aktywności.

wskazuje, że **MainActivity** jest **aktywnością aplikacji**

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:theme="@style/Theme.MyApplication">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

AndroidManifest.xml

Kluczowy **plik konfiguracyjny** każdej aplikacji. Zawiera informacje niezbędne dla systemu Android, takie jak:

- Deklaracja aktywności, usług, odbiorników zdarzeń (Activity, Service, BroadcastReceiver)
- Uprawnienia wymagane przez aplikację (np. dostęp do Internetu, aparatu)

Deklaracja aktywności.

wskazuje, że **MainActivity** jest **aktywnością aplikacji**

Oznacza, że inne aplikacje mogą uruchomić tę aktywność (np. przez **Intent**).

Definiowanie aktywności startowej

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:theme="@style/Theme.MyApplication">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

AndroidManifest.xml

Kluczowy **plik konfiguracyjny** każdej aplikacji. Zawiera informacje niezbędne dla systemu Android, takie jak:

- Deklaracja aktywności, usług, odbiorników zdarzeń (Activity, Service, BroadcastReceiver)
- Uprawnienia wymagane przez aplikację (np. dostęp do Internetu, aparatu)

Deklaracja aktywności.

wskazuje, że **MainActivity** jest **aktywnością aplikacji**

Oznacza, że inne aplikacje mogą uruchomić tę aktywność (np. przez **Intent**).

Określa, że **MainActivity** jest **główną aktywnością aplikacji**, czyli tą, która uruchamia się **jako pierwsza**

Definiowanie aktywności startowej

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:theme="@style/Theme.MyApplication">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

AndroidManifest.xml

Kluczowy **plik konfiguracyjny** każdej aplikacji. Zawiera informacje niezbędne dla systemu Android, takie jak:

- Deklaracja aktywności, usług, odbiorników zdarzeń (Activity, Service, BroadcastReceiver)
- Uprawnienia wymagane przez aplikację (np. dostęp do Internetu, aparatu)

Deklaracja aktywności.

wskazuje, że **MainActivity** jest **aktywnością aplikacji**

Oznacza, że inne aplikacje mogą uruchomić tę aktywność (np. przez **Intent**).

Określa, że **MainActivity** jest **główną aktywnością aplikacji**, czyli tą, która uruchamia się **jako pierwsza**

Definiowanie aktywności startowej

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:theme="@style/Theme.MyApplication">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Sprawia, że aktywność pojawia się w **launcherze telefonu** jako **ikona aplikacji.**

Cykl Życia

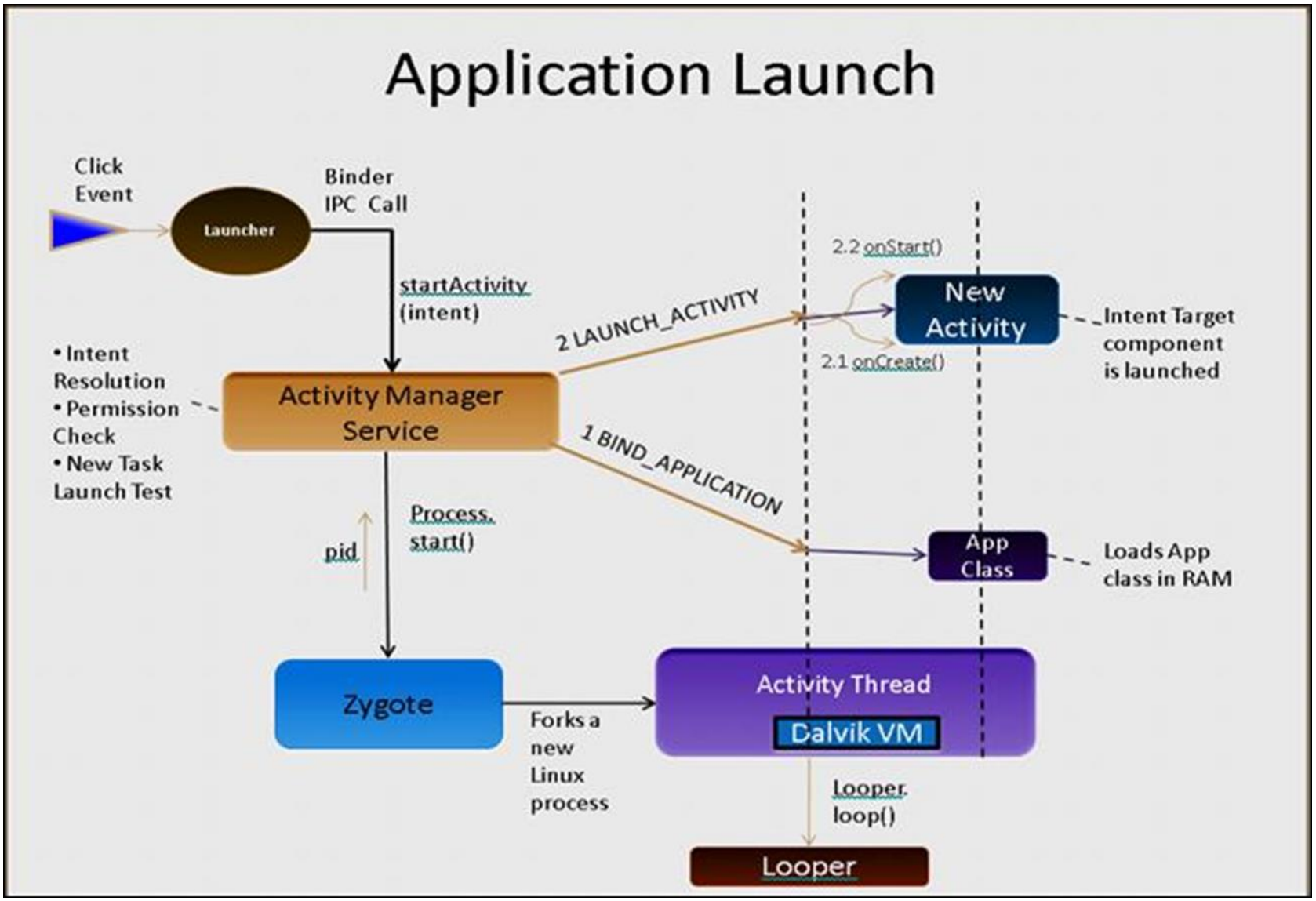
Każda aplikacja działa w swojej własnej **instancji maszyny wirtualnej.** W każdym momencie **kilka instancji** maszyny wirtualnej **może być aktywna.**

Aplikacja **nie kontroluje całkowicie** realizacji swojego cyklu życia.

OS może zakończyć każdy proces:

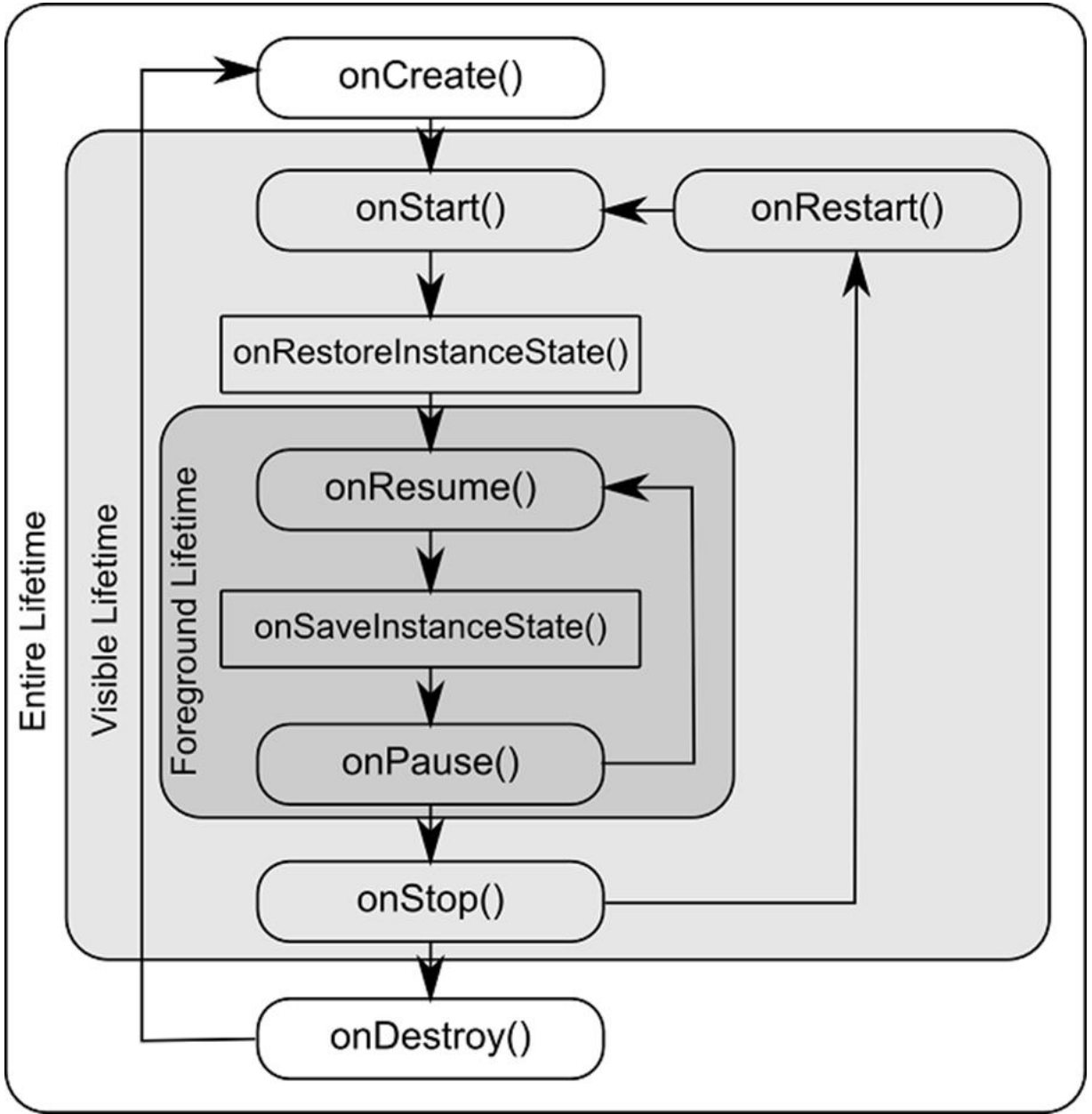
- Zasoby są krytycznie niskie
- Duża liczba działających aplikacji
- Aplikacja wymagająca bardzo dużych zasobów (energia, pamięć)

Cykl Życia


Application Launch
Click Event
Launcher
Binder IPC Call
startActivity (intent)
2 LAUNCH_ACTIVITY
2.2 onStart()
New Activity
2.1 onCreate()
Intent Target component is launched
• Intent Resolution
• Permission Check
• New Task Launch Test
Activity Manager Service
1 BIND_APPLICATION
App Class
Loads App class in RAM
pid
Process. start()
Zygote
Forks a new Linux process
Activity Thread
Dalvik VM
Looper. loop()
Looper

Uniwersytet Wrocławski

Cykl Życia Aktywności



Kontekst

APPLICATION

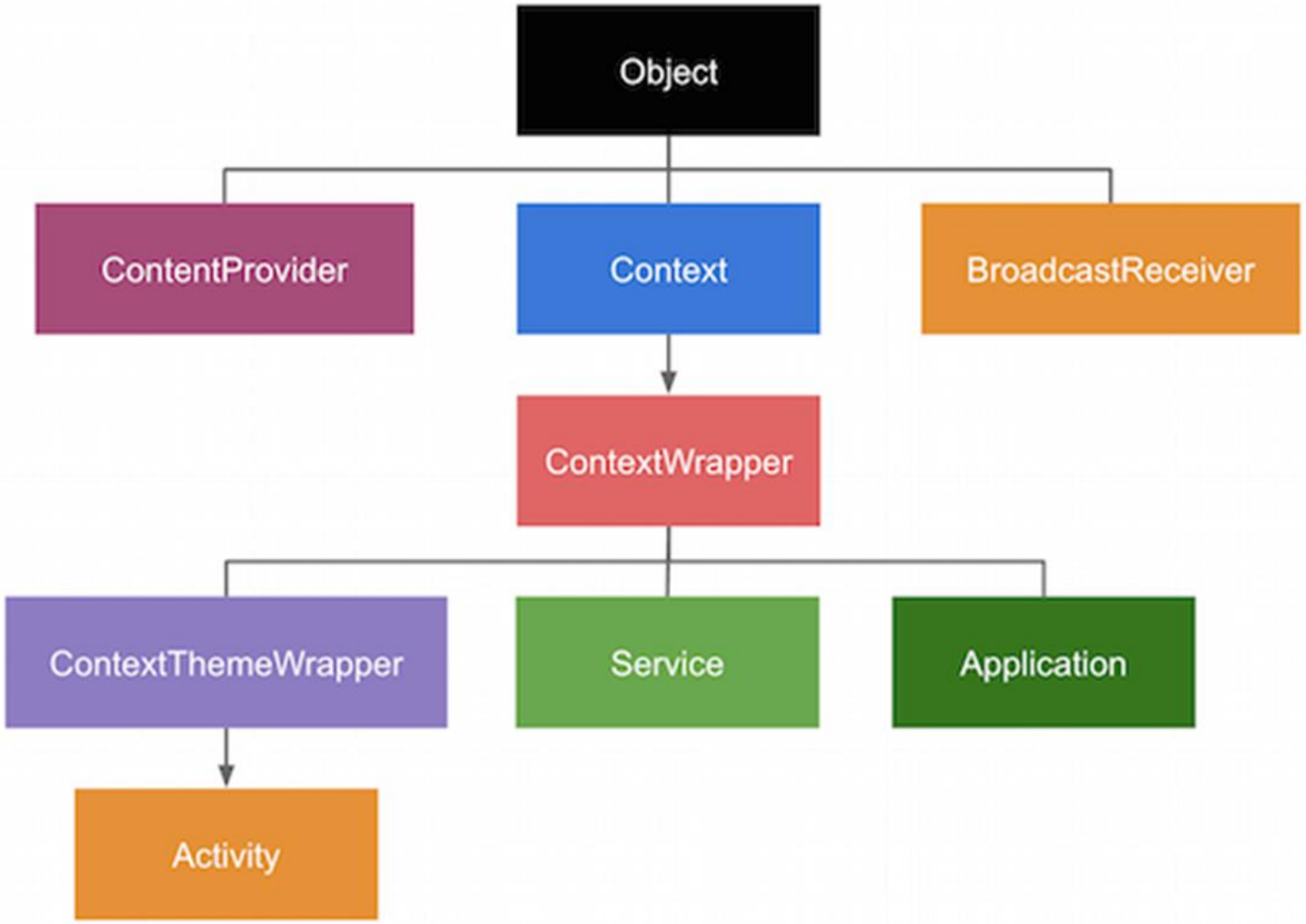
- **Globalny dostęp**: Application Context dostarcza **globalny dostęp do zasobów** aplikacji np. takich jak pliki zasobów (katalog res, klasa R)

- **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.

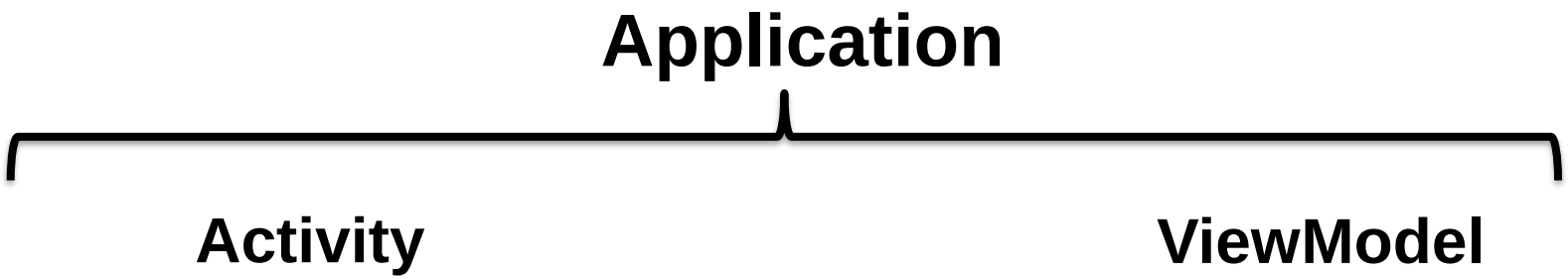
- **Bezpieczne do użycia w dłuższych operacjach**: Application Context jest bezpieczny do użycia w dłuższych operacjach, które mogą być wykonywane w tle, ponieważ **nie jest związany z cyklem życia UI** i nie powinien prowadzić do wycieków pamięci.

Uniwersytet Wrocławski

Kontekst



Kontekst



1. **Ograniczony do cyklu życia aktywności**: Kontekst aktywności ma ograniczony okres życia, który jest związany z cyklem życia aktywności.

Uniwersytet Wrocławski

Mechanizm 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 wykonanie 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 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 wykonanie 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 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 wykonanie 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 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 wykonanie 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 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 wykonanie 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.

Mechanizm 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 wykonanie 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.

Mechanizm Intent

- 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`

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

Mechanizm 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

```
val url = "https://netcat.uwr.edu.pl/"
val intent = Intent(Intent.ACTION_VIEW,
                Uri.parse(url)).apply{
                    addCategory(CATEGORY_BROWSABLE)
                }
context.startActivity(intent)
```