Systemy operacyjne na platformach mobilnych

Wykład 2

Grzegorz Jabłoński, Piotr Perek
Katedra Mikroelektroniki i Technik Informatycznych

Zagadnienia wykładu

- Interfejs użytkownika
  - Activity
  - Views
  - Resources
- Cykl życia aktywności
- Intencje

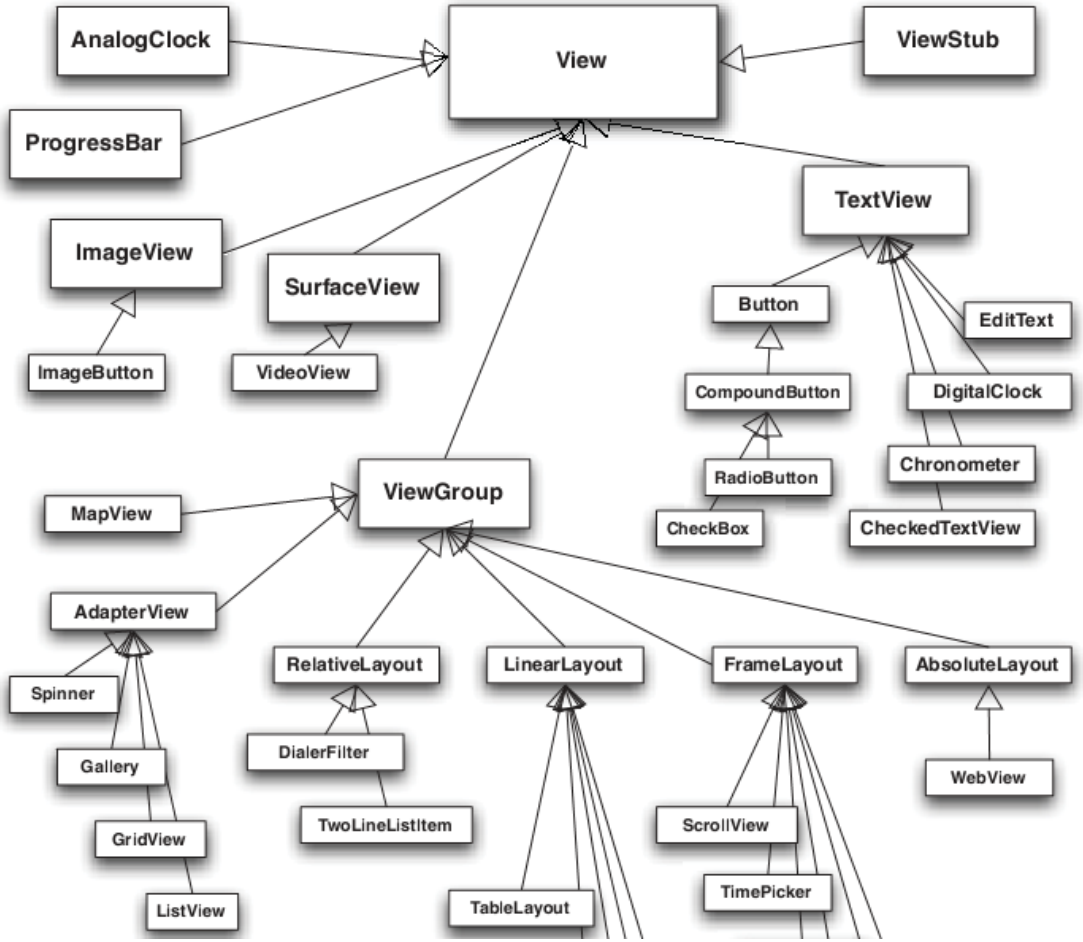
Aktywność (Activity)

- Jest podstawowym elementem aplikacji dla systemu Android
- Jest ściśle związana z ekranem, definiuje wygląd ekranu
- Bazuje na komponentach nazywanych widokami (views)
- Mechanizm przełączania aktywności bazuje na intencjach (intents)

Widoki (Views)

- Są obiektami, które użytkownik widzi i których używa do interakcji z aplikacją
- Obejmują:
  - układy kompozycji (layouts)
  - pola tekstowe (text elements)
  - przyciski (buttons)
  - obrazy (images)
  - i wiele innych…
- Korzystają z zasobów (resources) takich jak:
  - style
  - łańcuchy tekstowe
  - obrazy

Widoki (Views)



Intencje (Intents)

- Zapewniają prosty mechanizm do przełączania aktywności
- Opisują operacje do wykonania
  - akcja
  - dane (URI)

Sposoby wywołania aktywności

- Poprzez class name (jawne)
  - Dokładnie jedna aktywność może być dopasowana do intencji
  - Aktywność musi być w tym samym projekcie, co aktywność wywołująca
  - Intencja może przenosić dane do aktywności poprzez obiekt Bundle
- Poprzez URI (niejawne)
  - Więcej niż jedna aktywność może być dopasowana do intencji
  - Intencja może użyć class name lub URI aby określić aktywność
  - Aktywność nie musi być w tym samym projekcie, co aktywność wywołująca
  - Intencja może wysyłać dane do aktywności poprzez parametry URI lub obiekt Bundle
- Aktywność wywoływana jest metodą startActivity obiektu Context z intencją przekazaną jako argument.

Wywołania jawne

- Jawne wywołanie następuje, gdy wywołana zostaje konkretna aktywność, np. aktywność naszej aplikacji.
  - Jawne wywołanie wykonywane jest przez zdefiniowanie parametru Class lub ComponentName. ComponentName zawiera w pełni kwalifikowaną nazwę klasy składającą się z nazwy pakietu i nazwy klasy.

```
Intent(Context, Class)// konstruktor Intent
setComponent(ComponentName)// metoda Intent
setClass(Context, Class)// metoda Intent
```

```
startActivity(newIntent(this, MyActivity.class));
```

```
ComponentName component =
  newComponentName(
    MyActivity.class.getPackage().getName(),
    MyActivity.class.getName());
Intent intent = newIntent();
intent.setComponent(component);
startActivity(intent);
```

```
Intent intent = new Intent();
intent.setClass(this, MyActivity.class);
startActivity(intent);
```

Informacje w intencji

- Intencje oprócz wywołania aktywności, mogą także przenosić w sobie dane
- Dane przenoszone są w postaci obiektu klasy Bundle
- Bundle przypomina map<> z java – posiada klucze i wartości
- Do intencji obiekt Bundle wstawia się metodą putExtras, wyjmuje metodą getExtras (najczęściej po stronie uruchomionej przez intencję aktywności)

Dane intencji - Bundle

- W aktywności wywołującej:

```
Intent activityIntent = new Intent(this, NewActivity.class);
Bundle newActivityInfo = new Bundle();
newActivityInfo.put......(…);// putDouble, putString, etc.
activityIntent.putExtras(newActivityInfo);
startActivity(activityIntent);
```

- W aktywności wywołanej (NewActivity):

```
Intent intent = getIntent();
Bundle info = intent.getExtras();
```

Wywołania niejawne

- Niejawne wywołanie aktywności następuje w momencie, gdy platforma podejmuje decyzję, którą aktywności wywołać. Np. użytkownika interesuje jedynie nawiązanie połączenia telefonicznego, nie ma znaczenia czy użytkownik ma systemową aplikację telefoniczną, czy aplikację innej firmy, która obsłuży użytkownika.
- Niejawne wywołanie aktywności następuje w momencie, gdy w intencji nie jest wyspecyfikowany komponent (lub klasa aktywności).
- Intencja musi zawierać odpowiednie informacje, na podstawie których system zdecyduje, który komponent jest najlepszy do obsługi intencji.

Wywołania niejawne

```
case MENU_WEB_REVIEW:
        .......
        intent = new Intent(Intent.ACTION_VIEW, Uri.parse(link));
case MENU_MAP_REVIEW:
        .......
        intent = new Intent(Intent.ACTION_VIEW,
                Uri.parse("geo:0,0?q=" + location.getText().toString()));
case MENU_CALL_REVIEW:
        ........
        String phoneString = parsePhone(phone.getText().toString());
        intent = new Intent(Intent.ACTION_CALL,
                Uri.parse("tel:" + phoneString));
```

Wywołania niejawne

Action	URI	Description
`Intent.ACTION_CALL`	`tel:phone_number`	Opens the phone application and calls the specified number
`Intent.ACTION_DIAL`	`tel:phone_number`	Opens the phone application and dials (but doesn't call) the specified number
`Intent.ACTION_DIAL`	`voicemail:`	Opens the phone application and dials (but doesn't call) the voicemail number
`Intent.ACTION_VIEW`	`geo:latitude,longitude`	Opens the maps application to the specified latitude and longitude
`Intent.ACTION_VIEW`	`geo:0,0?q=street+address`	Opens the maps application to the specified address
`Intent.ACTION_VIEW`	`http://web_address`	Opens the browser application to the specified URL
`Intent.ACTION_VIEW`	`https://web_address`	Opens the browser application to the specified secure URL

Wyszukiwanie odbiorcy intencji

- W systemie Android obsługę żądań intencji oprócz Activity mogą realizować także BroadcastReceiveri Service
- Wymienione trzy komponenty ogłaszają swoje możliwości za pomocą elementu <intent-filter> w pliku AndroidManifest.xml
- <intent-filter> zawiera odpowiednie informacje, na podstawie których system zdecyduje, który komponent jest najlepszy do obsługi intencji
- Android przekształca każdy element <intent-filter> w obiekt IntentFilter

Wyszukiwanie odbiorcy intencji

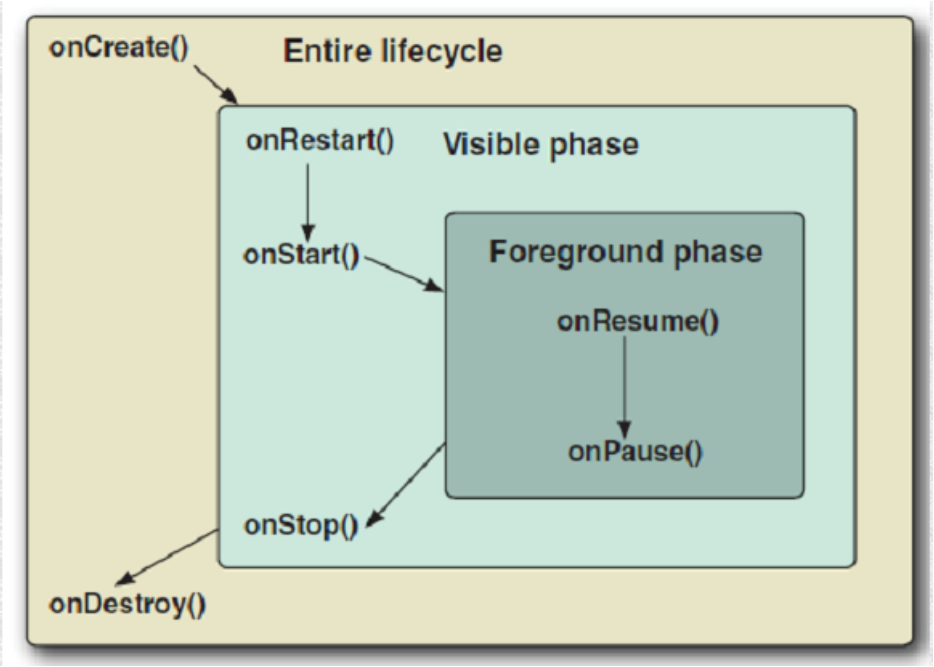
- Po zainstalowaniu pliku .apk Android rejestruje komponenty aplikacji w tym filtry intencji
- Posiadając rejestr filtrów intencji Android może odwzorować każde nie-jawne żądanie intencji na właściwy, zainstalowany obiekt Activity, BroadcastReceiver i Service.
- Dopasowanie następuje poprzez analizę:
  - akcji
  - typu danych
  - kategorii

Cykl życia aktywności

- Procesy umieszczane są na stosie.
- Procesy są związane z aktywnościami.
- Proces aktywności na pierwszym planie umieszczany jest na szczycie stosu.
- W momencie zmniejszania zasobów Android decyduje o tym, który proces zabić:
  - każdy proces nie obsługujący żadnej aktywności (ani usługi ani BroadcastReceiver) jest nazywany pustym procesem i jest pierwszym do usunięcia
  - każdy proces obsługujący aktywność działającą w tle jest następny w kolejce
  - każdy proces obsługujący widoczną aktywność, ale niebędącą na pierwszym planie, jest następny w kolejce
  - proces obsługujący widoczną aktywność działającą na pierwszym planie jest najważniejszy

Cykl życia aktywności

- W momencie tworzenia aktywności, przełączania na pierwszy plan, w tło i wyłączenia wywoływane są przez Androida odpowiednie metody Aktywności



Cykl życia aktywności

Method	Purpose
`onCreate()`	Called when the `Activity` is created. Setup is done here. Also provided is access to any previously stored state in the form of a `Bundle`.
`onRestart()`	Called if the `Activity` is being restarted, if it's still in the stack, rather than starting new.
`onStart()`	Called when the `Activity` is becoming visible on the screen to the user.
`onResume()`	Called when the `Activity` starts interacting with the user. (This method is always called, whether starting or restarting.)
`onPause()`	Called when the `Activity` is pausing or reclaiming CPU and other resources. This method is where you should save state information so that when an `Activity` is restarted, it can start from the same state it was in when it quit.
`onStop()`	Called to stop the `Activity` and transition it to a nonvisible phase and subsequent lifecycle events.
`onDestroy()`	Called when an `Activity` is being completely removed from system memory. This method is called either because `onFinish()` is directly invoked or the system decides to stop the `Activity` to free up resources.

Layouty

- W systemie android układ ekranu definiowany jest za pomocą obiektów ViewGroup i LayoutParams.
  - ViewGroup jest widokiem, który zawiera inne widoki
  - Każdy typ ViewGroup posiada LayoutParams, które opisują sposób rozmieszczenia widoków podrzędnych (komponentów)
  - Komponenty rysowane są na ekranie w kolejności, w jakiej znajdują się w drzewie layoutu

Typy layoutów

- AbsolutLayout – pozwala określić dokładne współrzędne x i y dla poszczególnych komponentów
- FrameLayout – wszystkie elementy są wyrównane do lewego górnego rogu (przysłaniają się)
- LinearLayout – kolejne elementy są układane od góry do dołu (vertical) lub od lewej do prawej (horizontal)
- RelativeLayout – pozwala określić wzajemne położenie elementów względem siebie
- TableLayout – elementy są układane w postaci macierzy (w kilku wierszach i kolumnach)

Typy zasobów

- res/anim — reprezentacja XML animacji 'ramka po ramce‘
- res/drawable — .png, .png, i .jpg obrazy
- res/layout — reprezentacja XML obiektów widoków
- res/values — reprezentacja XML napisów, kolorów, styli, rozmiarów i tablic
- res/xml — zdefiniowane przez użytkownika pliki XML aplikacji kompilowane do postaci binarnej
- res/raw — pliki niekompilowane dodawane do aplikacji