LUKÁŠ NOVÁK

KMI / TMA

TVORBA MOBILNÍCH APLIKACÍ

4. SEMINÁŘ | 7.10.2020

ZS 2020/2021 | STŘEDA 15:00-17:30

OBSAH SEMINÁŘE

PŘEDÁVÁNÍ DAT MEZI AKTIVITAMI, SEZNAMY A ADAPTÉRY

PŘEDÁVÁNÍ DAT

AKTIVITA <-> AKTIVITA

- nemáme k dispozici konstruktor aktivity, pomocí kterého bychom ji mohli předat data
- řešení nabízí **Intent**, kterým určujeme, kterou aktivitu má systém spustit
- při vytváření instance přidáme data, která chceme předat

PŘEDÁVÁNÍ DAT

ULOŽENÍ DAT

- Intent má (přetížené) metody pro předání různých datových typů
  - **Intent.putExtra(String, String)**
  - **Intent.putExtra(String, int)**
  - **Intent.putExtra(String, float)**
  - ...
- jako první argument předáme klíč pro následné získání hodnoty

PŘEDÁVÁNÍ DAT

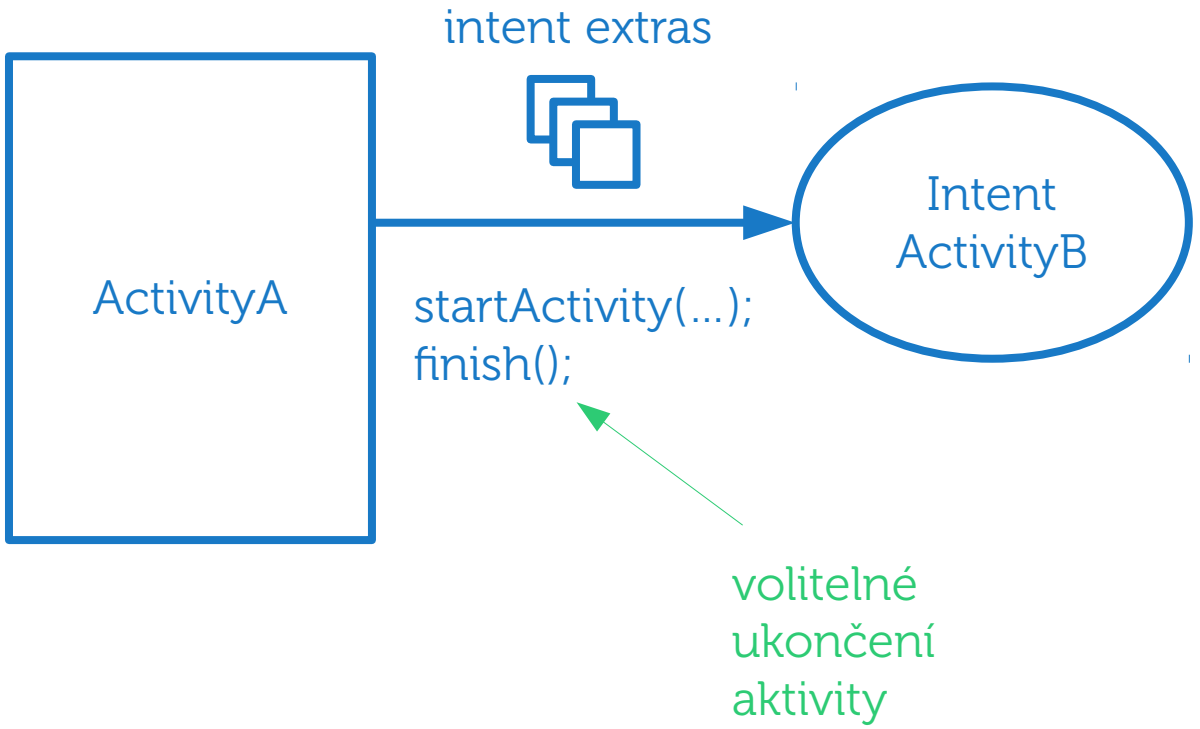
ZÍSKÁNÍ DAT

- ve spouštěné aktivitě získáme instanci Intentu pomocí metody **getIntent()**
- hodnoty získáme pomocí metod
  - **Intent.getStringExtra(String)**
  - **Intent.getIntExtra(String, int)**
  - **Intent.getFloatExtra(String, float)**
  - ...
- některé metody vyžadují defaultní hodnotu pro případ, že daný klíč neexistuje

SPOUŠTĚNÍ AKTIVIT

NEČEKAJÍCÍ SPUŠTĚNÍ

1/1



SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

- **volající aktivita** bude „čekat“ na data od **volané aktivity**
- pro získání dat z **aktivity B** v **aktivitě A** tedy není nutné vytvářet novou **aktivitu A** (ta zůstává v pozadí „pod“ **aktivitou B**)

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

- ~~startActivity(Intent)~~
  startActivityForResult(intent, requestCode)
  - **intent**: standardní intent popisující aktivitu pro spuštění
  - **requestCode**: určuje identifikátor tohoto požadavku pro následné získání dat z volané aktivity

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

```
public static final int REQUEST_EDIT = 1;

@Override
protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    Log.d(TAG, "onCreate()");

    setContentView(R.layout.activity_main);

    mInsertButton = (Button) findViewById(R.id.insert_button);
    mInsertButton.setOnClickListener((v) -> {
            Intent i = new Intent(MainActivity.this, EditActivity.class);
            startActivityForResult(i, REQUEST_EDIT);
    });

    // initialize other Views

}
```

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

- předání dat zpět před ukončením aktivity
  - zavoláme **setResult(resultCode)** nebo **setResult(resultCode, data)**
    - **resultCode**: kód výsledku (jakýkoli int)
    - **data**:
      - pokud nestačí **resultCode**, vytvoříme instanci **Intentu** pomocí konstruktoru bez parametrů
      - přidáme **Intentu** data, které chceme předat zpět

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

```
Intent resultData = new Intent();
resultData.putExtra("NAME", "John");
resultData.putExtra("SURNAME", "Doe");
setResult(MainActivity.RESULT_EDIT_OK, resultData);
finish();
```

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

- ve volané aktivitě přepíšeme metodu **onActivityResult(requestCode, resultCode, data)**
  - **requestCode**: odpovídá číslu předanému v startActivityForResult(intent, **requestCode**)
  - **resultCode**: odpovídá číslu předanému v setResult(**resultCode**, data)
  - **data**: předaný Intent metodě setResult setResult(resultCode, **data**)

SPOUŠTĚNÍ AKTIVIT

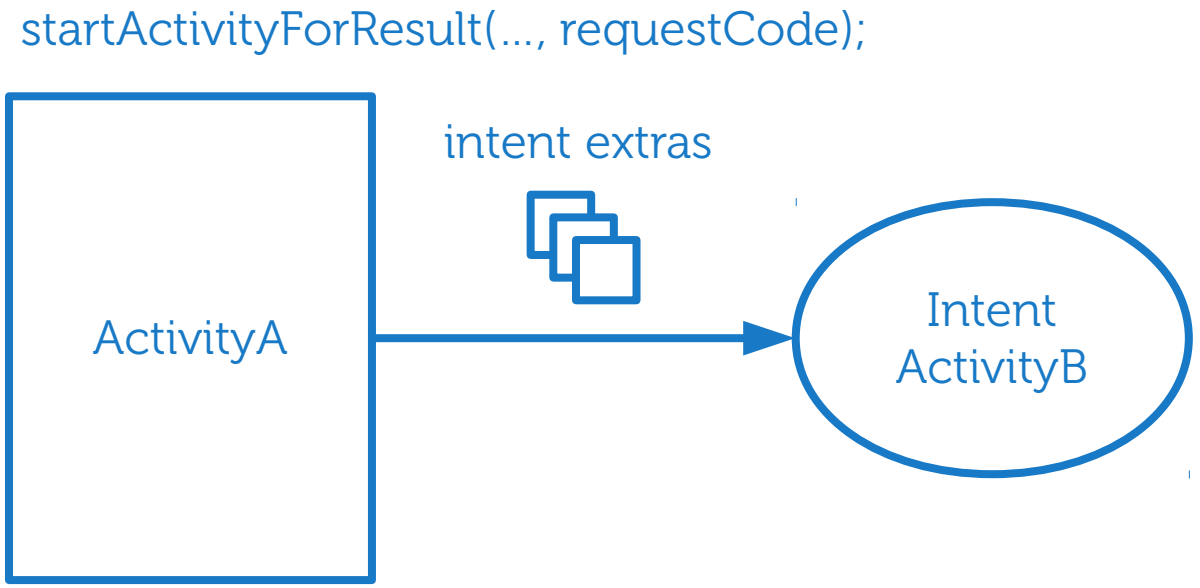
ČEKAJÍCÍ SPUŠTĚNÍ

```
public static final int REQUEST_EDIT = 1;
public static final int RESULT_EDIT_OK = 0;

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == REQUEST_EDIT) {
        if (resultCode == RESULT_EDIT_OK) {
            // do something
        }
    }

}
```

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

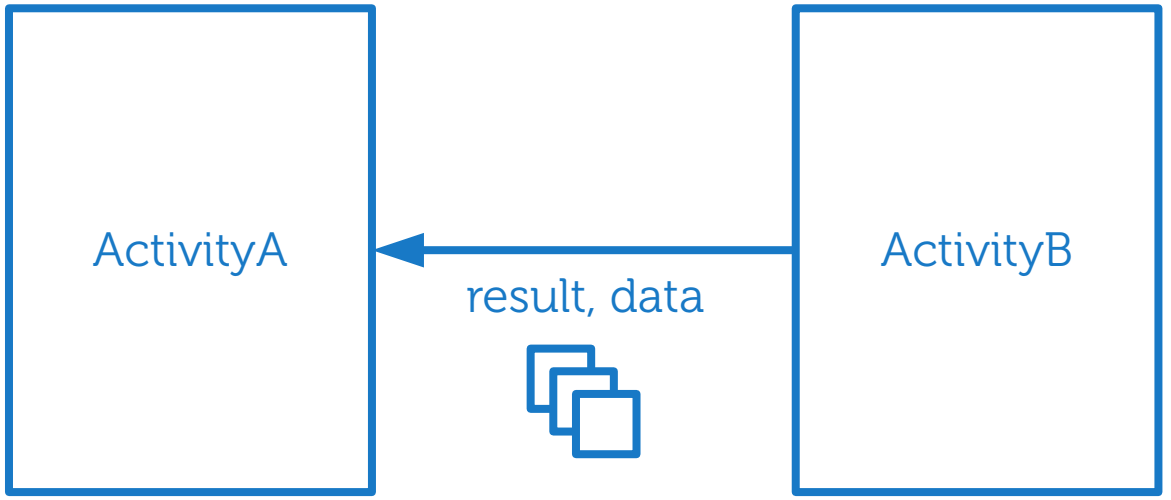
1/3



SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

2/3




```
setResult(result, data);
finish();
```

SPOUŠTĚNÍ AKTIVIT

ČEKAJÍCÍ SPUŠTĚNÍ

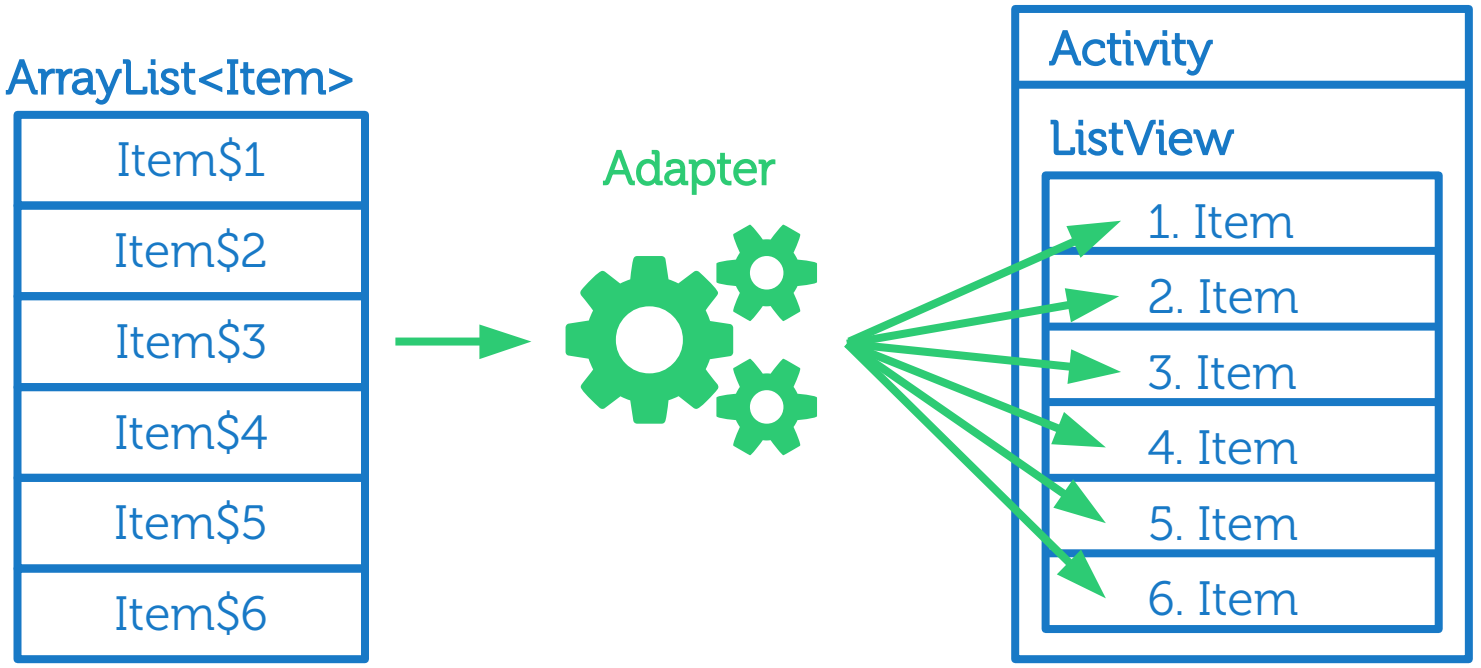
3/3

onActivityResult(requestCode, resultCode, data)

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- View pro seznam zobrazující data v kolekci
- data nepředáváme přímo **ListView**, ale prostředníkovi ~ **Adapter**



POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- **Adapter**
  - abstraktní třída, nelze vytvořit instanci
  - stará se o zobrazení prvků v seznamu
- **ArrayAdapter<T>**
  - implementace třídy Adapter pro použití s daty uloženými v kolekci **List<T>**
  - pro jednoduché použití je možné použít jeden z konstruktorů
  - pro pokročilé použití se dědí

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- **ArrayAdapter<T>(context, resource, textViewResourceId, objects)**
  - **context**: předání kontextu ~ aktivity (this)
  - **resource**: layout každého prvku v seznamu
  - **textViewResourceId**: identifikátor TextView v layoutu předaného v resource, do kterého adaptér vloží textovou reprezentaci (toString()) objekty z kolekce objects
  - **objects**: kolekce obsahující objekty pro zobrazení v ListView

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

» **resource**: layout každého prvku v seznamu

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/text"
    android:padding="16dp" />
```

textViewResourceId

resource

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- objects:
  kolekce objektů,
  viz příklad
  TodoItem:

```
package cz.upol.inf.kma.tmatodo.model;

public class TodoItem {

    private String mTitle;
    private String mContent;

    public TodoItem(String title, String content) {
        mTitle = title;
        mContent = content;
    }

    public String getTitle() { return mTitle; }

    public void setTitle(String title) { mTitle = title; }

    public String getContent() { return mContent; }

    public void setContent(String content) { mContent = content; }

    @Override
    public String toString() { return mTitle; }

}
```

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- propojení **ListView** a **Adapter**
  - **ListView.setAdapter(Adapter)**
- následná práce s kolekcí v adaptéru
  - **Adapter.add(T object)**
  - **Adapter.remove(T object)**
  - **Adapter.getItemAtPosition(int position)**
  - ...
- při změně kolekce přímo je nutné po každé změně aktualizovat adaptér pomocí metody
  - **Adapter.notifyDataSetChanged()**

POKROČILÉ VIEWS

LISTVIEW / ADAPTER

- reakce na kliknutí do seznamu
  - ~~ListView.setOnClickListener~~
  - ListView.setOnItemClickListener(...)
  - **OnItemClickListener.onItemClick (parent, view, position, id)**
    - **parent** ~ ListView
    - **position**: pozice prvku, na který bylo kliknuto

ÚKOL 4. SEMINÁŘE

SEZNAM TODO POLOŽEK

1) Do MainActivity přidat kolekci (ArrayList) TODO položek. TODO položka bude reprezentována třídou TodoItem se složkami **title** a **content**.
2) Vytvořit **ArrayAdapter** a propojit tak kolekci **TodoItem** a seznam **ListView**. Resource layout se bude skládat z jednoho **TextView**.
3) Implementovat obousměrnou komunikaci mezi **MainActivity** a **EditActivity** (přejmenovaná InsertActivity).
4) Klik na Insert spustí **EditActivity**. Vyplněná data předá zpět do **MainActivity**, která vytvoří TodoItem a přidá do seznamu.
5) Klik na položku v seznamu spustí **EditActivity** s předvyplněným titulkem a obsahem. Vyplněná data předá zpět do **MainActivity**, která změní daný **TodoItem** a aktualizuje seznam.

Hint) Mezi aktivitami budeme předávat data **id**, **content** a **title**. Pro vkládání nové položky bude id = -1. V takovém případě nebudeme plnit EditText v EditActivity. Podle předaného id zpět poznáme, zda vkládat novou položku nebo editovat stávající.

OTÁZKY

PTEJTE SE!