Week 7 - oefening 1: GridView

Doelstelling

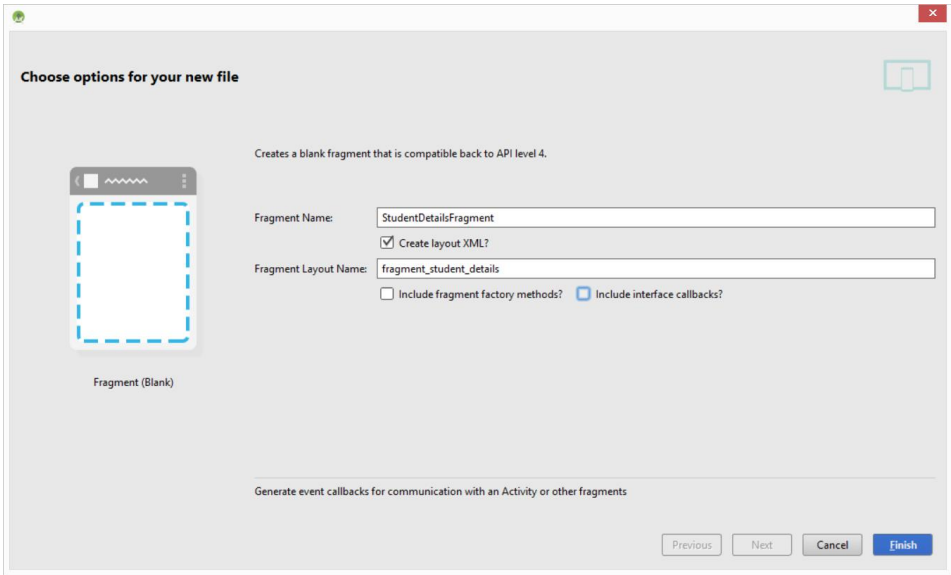
Gebruik leren maken van GridView, Loader, Adaptar en andere gerelateerde layout(s).

Voorbereiding

Deze labo-opgave werkt verder op de labo-opgave 2 van week 6. *Tip: maak een backup voordat u met deze oefening verder gaat!*

StudentDetailsFragment

Voeg een nieuw fragment 'StudentDetailsFragment' toe (type Blank Fragment).



StudentDetailsFragment - Layout

In deze Fragmentklasse StudentDetailsFragment overschrijf je de onCreateView-methode waarin de layoutfile **fragment_student_details** wordt gebruikt. In de layout-file plaatsen we:

- TextView waar de totale score van de student wordt aangegeven
- GridView met 2 kolommen. (Het aantal rijen wordt automatisch bepaald)
- TextView om de diplomagraad weer te geven (uitbreiding)

Meer info over een Gridview vindt u hier:

http://developer.android.com/guide/topics/ui/layout/gridview.html

Haal in de onCreateView-methode alle 3 views op zodat deze later in verschillende methodes kunnen benaderd worden.

Voeg zoals in voorbije opgaves de default-constructor en de static newInstance-methode toe.

Welke student?

Uiteraard is het bedoeling dat we enkel informatie van één specifieke student gaan weergeven. De klasse StudentAdmin heeft een static-methode dat via het e-mailadres de gegevens van de bijhorende student kan opvragen.

Van wie moet het e-mailadres komen?
Voeg aan de newInstance-methode minimaal één parameter voor het e-mailadres toe. Je kan ook naam en voornaam als parameters toevoegen.

Pas de code op verschillende plaatsen aan zodat e-mailadres, naam en voornaam getoond worden:

- StudentsFragment -> onListItemClick-methode
- StudentsFragment rapporteert terug naar Activity (werkwijze zie labo week 5)
- StudentActivity stuurt info door naar StudentDetailsFragment via de newInstance-methode
- StudentDetailsFragment toont naam en voornaam in de onCreateView-methode

GridView

Het opvullen van de gridview gebeurt op analoge wijze als het opvullen van een listview. Voor een gedetailleerde beschrijving: zie labo week 6 oefening 2.

Volgende stappen dienen doorlopen te worden:

- Breid de klasse **Contract** uit:

```
public interface ModulePuntColumns extends BaseColumns {
    public static final String COLUMN_MODULE_NAAM = "module_naam";
    public static final String COLUMN_MODULE_SCORE = "module_score";
    public static final String COLUMN_MODULE_STUDIEPUNTEN = "module_studiepunten";
}
```

- Maak een klasse **ModulePuntLoader** in de package be.howest.nmct.loader dat verantwoordelijk is om de data asynchroon op te halen, en een Cursor (MatrixCursor) af te

leveren.

- Maak een extra layout-file '**cel_module**' aan, verantwoordelijk voor het design van één cel in de grid.

- Maak in de klasse StudentDetailsFragment een adaptarklasse '**ModulePuntAdaptar**' aan. Deze klasse staat in voor het visualiseren van de data afkomstig van de cursor. In de 'bindView'-methode gebruik je de views gedefinieerd uit de layoutfile 'cel_module'.

- Zorg ervoor dat StudentDetailsFragment kan communiceren met de **LoaderManager**. Dit doe je door de klasse de interface 'LoaderManager.LoaderCallBacks' te laten implementeren. Werk de drie interface-methodes uit.

- Werk tenslotte de methode **onActivityCreated** in StudentDetailsFragment uit: hier wordt de adaptar geïnitialiseerd en aan gridview gekoppeld. Tenslotte wordt de loader geactiveerd via het commando getLoaderManager().initLoader(…)

Test alles uit!

Afprinten graad diploma

We printen onderaan de diplomagraad af. Hiervoor hebben we de totale score nodig. Deze info hebben we niet rechtstreeks. We kunnen wel de totale score reconstrueren met alle opgevraagde data (aanwezig in de cursor).

```
public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {
    mAdapter.swapCursor(cursor);
    printDiplomaGraad(cursor);
}

private void printDiplomaGraad(Cursor cursor) {
    ArrayList<Double> scoresModules = new ArrayList<>();
    ArrayList<Integer> studiepuntenModules = new ArrayList<>();

    int aantalRijen = cursor.getCount();
    for (int i = 0; i < aantalRijen; i++){
        cursor.moveToPosition(i);
        int colnr1 = cursor.getColumnIndex(Contract.ModulePuntColumns.COLUMN_MODULE_STUDIEPUNTEN);
        studiepuntenModules.add(cursor.getInt(colnr1));

        int colnr2 = cursor.getColumnIndex(Contract.ModulePuntColumns.COLUMN_MODULE_SCORE);
        scoresModules.add(cursor.getDouble(colnr2));
    }

    int iTotaalStudiepunten = 0;
    for (int aantalSP : studiepuntenModules) iTotaalStudiepunten += aantalSP;

    double dTotaleScore = 0;
    for (int iTeller = 0; iTeller < scoresModules.size(); iTeller += 1) {
        double dGewicht = (double) studiepuntenModules.get(iTeller) / iTotaalStudiepunten;
        dTotaleScore += (scoresModules.get(iTeller) * dGewicht);
    }

    textViewDetailGraadDiploma.setText(Student.DIPLOMAGRAAD.getDiplomagraad((float)dTotaleScore).getOmschrijving());
}
```