**JOURNAL DE BORD**

**POUR NOTER SA JOURNEE**

***RAPPORT DE PROGRAMMATION ET INTERFACES ANDROID***

***LICENCE 3 MIME***

**Enseignant : Guillaume Besacier**

***DEVELOPPEUSES***

KOUAOUCI Asma Amina

MOHAMMEDI Kahina

KAIS Tiziri

Table des matières

1. INTRODUCTION 3

2. INITIALISATION DU PROJET 3

3. ARCHITECTURE DE L’APPLICATION 4

4. Activités et Fragments 5

4.1. Main Activity  5

4.1.1. Personnaliser le Navigation Drawer Activity 5

4.2. MyDayFragment (FragmentList) 6

4.3. CriteriaDayActivity (Activity) 6

4.4. NewCriteriaDayActivity (Activity) 6

4.5. DaysHistoryFragment (FragmentList) 7

4.6. DetailHistoryActivity (Activity) 7

4.7. GrapheFragment (Fragment) 7

4.8. AproposFragment (Fragment) 7

5. Le Model 7

6. Stockage (Base de données)  8

7. Personnalisation de la ListView  9

8. Test et correction de bugs 10

# INTRODUCTION

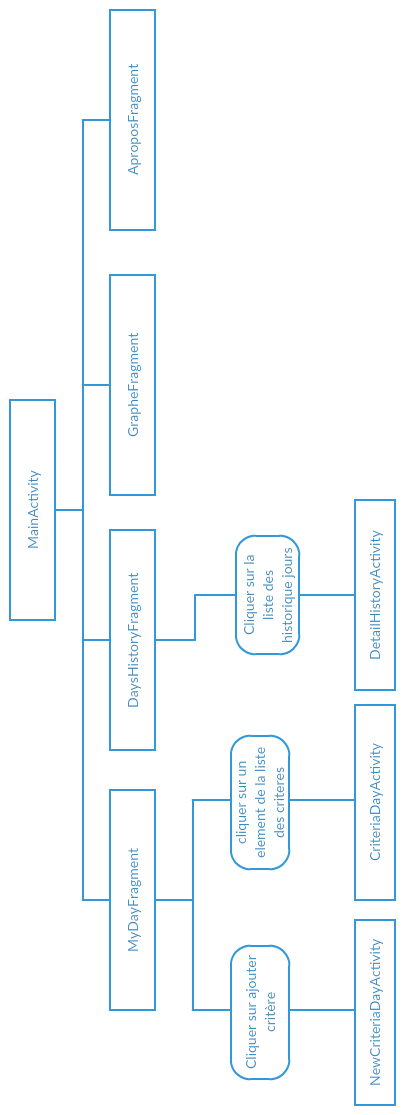
« *Journal de bord pour noter sa journée*» est une application qui permet à l’utilisateur de noter ses journées sur une base de critères (Santé, Études, Nourriture, Sommeil, Sport, Famille, ect …) afin qu’il puisse identifier les critères négatifs et pouvoir améliorer ses journées.

# **I**NITIALISATION DU PROJET

Pour notre application on a choisi d’utiliser «***Navigation Drawer Activity****»* comme activité principale, cela nous a permis d’avoir un menu pour naviguer dans l’application.

On a supprimé les éléments dont on a pas besoin dans notre application (l’item settings dans main.xml, le FloatingActionButton dans app\_bar\_main.xml

# ARCHITECTURE DE L’APPLICATION



# Activités et Fragments

# Main Activity

C’est l’activité principale qui nous permet la navigation entre les éléments du menu Drawer en appelant la méthode onNavigationItemSelected.

# Personnaliser le Navigation Drawer Activity

On a modifié les éléments du menu (images/textes/id) dans les fichiers « act*ivity\_main\_drawer.xml, nav\_header\_main.xml* » afin d’avoir le menu de notre application

****

# MyDayFragment (FragmentList)

Ce fragment est affiché lorsqu’on clique sur l’élément ma journée dans le menu,

Il est de type *ListFragment, il contient la liste des critères de la journée, quelques critères sont ajouté par défaut (Santé, Culture, Sport), et la note* (RatingBar de type Indicateur) *de la journée calculé a partir de tous les notes des critères.*

# CriteriaDayActivity (Activity)

Cette activité affiche le détail d’un critère journée qui contient la description (EditText) et la note (RatingBar) du critère.

Aussi trois boutons sont disponibles :

* Sauvegarder : permet de mettre a jour le critère dans la base de donnée.
* Supprimer : permet de supprimer le critère de la base de donnée
* Annuler : permet de fermer l’activité.

# NewCriteriaDayActivity (Activity)

Cette activité affiche un formulaire d’ajout d’un nouveau critère, elle contient le nom (EditText), la description (EditText) et la note (RatingBar) du critère.

Aussi deux boutons sont disponibles :

* Sauvegarder : permet d’ajouter le critère dans la base de donnée
* Annuler : permet de fermer l’activité.

# DaysHistoryFragment (FragmentList)

Ce fragment est affiché lorsqu’on clique sur l’élément Historique dans le menu,

Il est de type ListFragment, il contient la liste de tous les journées d’utilisateur,

Lorsqu’on clique sur un élément de la liste le détail de la journée est affiché.

# DetailHistoryActivity (Activity)

Cette activité permet d’afficher le détail d’une journée, elle contient un EditText avec le nom et description de tous les critères dans un format Html, et le rating de la journée.

# GrapheFragment (Fragment)

Ce fragment est affiché lorsqu’on clique sur l’élément Graphe dans le menu, il contient le graphe des notes de toutes les journées.

Pour permettre d’afficher ce graphe, on a utilisé une librairie tierce « ***graphview***»,

Afin d’ajouter la libraire dans le projet, on a ajouter la ligne « **compile 'com.jjoe64:graphview:4.2.1'**» dans le fichier **« build.gradle**» dans la section « **dependencies** ».

# AproposFragment (Fragment)

Ce fragment est affiché lorsqu’on clique sur l’élément A propos dans le menu, il contient un EditText avec les information sur les développeuses, l’enseignant dans un format Html.

On peut aussi envoyé un mail aux créateurs de l’application en cliquant sur le bouton mail.

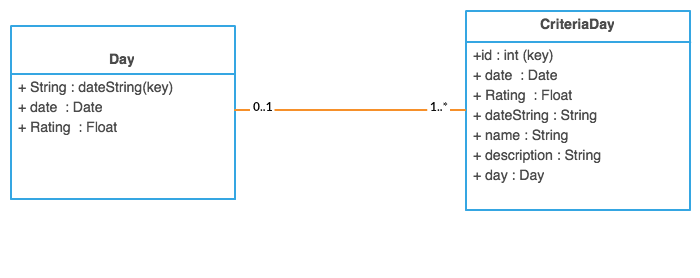
# Le Model

On a deux classes modèles :

* Day (Journée)
* CriteriaDay (Critère journée),

Une journée peut avoir plusieurs critères, et chaque critère n’a qu’une seule journée,

C’est une relation de one to many.

****

# Stockage (Base de données)

Pour la base de données, trois classe sont crées :

* **DBHelper** qui étends **SQLiteOpenHelper**: Cette classe permet la création des deux tables **Day** et **CriteriaDay** dans sa méthode **onCreate**,

On surcharge aussi **onUpgrade** pour permettre la mise à jour la base de donnée dans des prochaines versions.

* **DayDAO :** Cette classe permet la gestion de la table **Day,** pour celaon a implémentées plusieurs méthodes :
  + **createDay :** permet la création d’une journée « **Day**» dans la table.
  + **insertDay :** permet l’insertion d’une nouvelle journée dans la table.
  + **deleteDay :** permet la suppression d’une journée de la table (remarque : cette méthode n’est pas utilisé mais on l’a gardé pour l’utilisé dans la prochaine version qui permet la gestion de l’historique**).**
  + **getAllDays :** retourne une liste de toutes les journées**.**
  + **getDayById :** retourne un jour « **Day**»  par l’identifiant Id.
* **CriteriaDayDAO :** Cette classe permet la gestion de la table **CriteriaDay,** pour celaon a implémentées plusieurs méthodes :
  + **createCriteriaDay:** permet la création d’une journée « **CriteriaDay**» dans la table.
  + **getCriteriaDayOfDay :** retourne une liste de CriteriaDay pour un jour donnée Day (Cette méthode est appelé dans le fragment myDayFragment et dans l’activité DetailHistoryActivity).
  + **getCriteriaDay :** retourne un jour «**CriteriaDay**»  par l’identifiant Id.
  + **insertCriteriaDay :** permet l’insertion d’un nouveau critère jour «**CriteriaDay**»  dans la table.
  + **deleteCriteriaDay :** permet la suppression d’un critère jour «**CriteriaDay**»  de la table.
  + **updateCriteriaDay :** permet la mise à jour du «CriteriaDay» dans la table.

# Personnalisation de la ListView

Afin de personnaliser la ListeView, on a crée une classe **CustomAdapter,** cette dernière est utilisée dans deux FragmentList « **MyDayFragment**» pour la liste des critère et « **DaysHistoryFragment**» pour afficher l’historique journées.

Cette classe a besoin en entrée :

* D’une liste Critères « **CriteriaDay**»  ou journée « **Day**» pour avoir le nom et la note.
* D’un type de liste « **MyDayFragment**» ou « **DaysHistoryFragment** pour l’affichage de l’activité lors du clique.
* Du FragmentList pour récupérer l’activité parente et l’utilisé pour l’affichage de la nouvelle activité.

Le Layout **Custom\_adapter.xml** contient **:**

* Une image : pour l’image on a choisit d’utiliser un smiley qui définisse la note
  +  si la note est inferieur à 1.
  + si la note est entre 1 et 2.
  + si la note est entre 2 et 3.
  +  si la note est entre 3 et 4.
  +  si la note est entre 4 et 5.
* Un TextView : le nom du critère ou la date de la journée.

# Test et correction de bugs

Durant tout le processus du développement ainsi lors du test de l’application par nous même ou quelques membre de nos famille,

On a trouvé quelques bugs dans l’application qu’on a veillé à résoudre :

Parmi ces bugs trouvés, on a :

* **La suppression de tous les critères d’une journée**:
  + On a remarque que lorsque un utilisateur supprime tous les critère d’une journée, les critère se réinitialise avec les critères par défaut, pour remédier a ce bug, on a ajoutée un test lors de la suppression pour que la suppression ne soit possible que si le nombre de critère est supérieur à 1.
* **Lors de la mise à jour où la suppression d’un critère la liste ne se mettait pas a jour au retour de l’activité :**
  + En effet, l’activité ne se mettait pas a jour, pour corrigé ce bug, on a utilisé **startActivityForResult** (avec le requestCode 10001 pour MyDayFragment) pour l’affichage de l’activité et on a surcharger **onActivityResult** pour le retour et mettre a jour le fragment.