

Développement Android

Laboratoire n°3

Interactions avec l'utilisateur

Approche MVC

Introduction

Ce laboratoire est constitué de plusieurs manipulations et questions théoriques destinées à appréhender la mise en place des différents composants *Android* de base pour la réalisation d'une *Activité* proposant un formulaire permettant d'éditer les informations d'une personne.

Manipulations

1. Mise en place

Pour ce laboratoire, vous allez partir du template d'une application vide (*Projet Android template*) disponible sur Cyberlearn auquel vous intégrerez les éléments que nous vous fournissons au sein d'un fichier *.zip*, à savoir :

- *Person.kt* : contient la classe abstraite *Person* ainsi que ses deux sous-classes *Worker* et *Student*. Nous vous avons également mis à disposition deux instances statiques d'exemple : *exampleWorker* et *exampleStudent* contenant des données illustratives. Le diagramme *UML* correspondant est disponible dans les slides de présentation ;
- *strings.xml* : contient la définition de textes et de tableaux de valeurs utilisables dans ce laboratoire. Il s'agit d'une proposition pour vous faire gagner du temps, sentez-vous libre de l'adapter ;
- *cake.xml* : image vectorielle représentant un gâteau d'anniversaire.

Attention : Si vous n'utilisez pas le projet template fourni comme point de départ, vous devrez potentiellement ajuster le nom de package au niveau du fichier kotlin.

2. Développement de la vue

Nous vous demandons à présent de réaliser l'interface graphique de notre application. Il s'agit de concevoir le fichier *layout* intégrant et mettant en page tous les textes, les widgets et les différents éléments de notre interface graphique. Vous trouverez sur la Fig. 1 une proposition de mise en page, nous vous demandons de réaliser votre UI à l'aide d'un *ConstraintLayout*. Vous veillerez à ne pas

imbriquer un trop grand nombre de *layouts* pour ne pas péjorer les performances, un bonus vous sera accordé si vous arrivez à réaliser votre interface graphique avec un seul *ConstraintLayout*. Attention, nous parlons ici du contenu uniquement. Nous ne comptons pas ici le *layout* parent englobant l'AppBar et le *layout* gérant le contenu. Cette approche est tout à fait acceptable et même conseillée ici.

The figure shows three examples of a person editing form interface, each with a purple header 'Labo 3'. The forms are color-coded to highlight different sections: green for base data, orange for specific data, and red for common complementary data.

- Form 1 (General Person):**
 - DONNÉES DE BASE (Green):** Nom, Prénom, Date de naissance, Nationalité (Sélectionner), Occupation (Etudiant, Employé).
 - DONNÉES COMPLÉMENTAIRES (Red):** Adresse e-mail, Commentaires, Annuler, OK.
- Form 2 (Student):**
 - DONNÉES DE BASE (Green):** Nom (Dreher), Prénom (Matthias), Date de naissance (8 avr. 1998), Nationalité (Allemande), Occupation (Etudiant, Employé).
 - DONNÉES SPÉCIFIQUES (ÉTUDIANTS) (Orange):** Ecole/Université (HEIG-VD), Année du diplôme (2023).
 - DONNÉES COMPLÉMENTAIRES (Red):** Adresse e-mail (m.dreher@email.com), Commentaires (Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.), Annuler, OK.
- Form 3 (Employee):**
 - DONNÉES DE BASE (Green):** Nom (Devost), Prénom (Chantal), Date de naissance (12 juin 1996), Nationalité (Suisse), Occupation (Etudiant, Employé).
 - DONNÉES SPÉCIFIQUES (EMPLOYÉ) (Orange):** Entreprise (HEIG-VD), Secteur (Sciences), Expérience (années) (2).
 - DONNÉES COMPLÉMENTAIRES (Red):** Adresse e-mail (c.devost@email.com), Commentaires, Annuler, OK.

Figure 1 - Exemple de l'interface graphique permettant d'éditer une Personne. Les zones en couleur servent à mettre en évidence les différentes parties du formulaire et ne doivent pas forcément être reproduites (en vert : données de base d'une personne, en orange : particularités d'un employé ou d'un étudiant, en rouge les champs complémentaires communs).

Votre interface devra permettre d'éditer des *Person*, soit des *Worker* ou des *Student*, en mettant à disposition les widgets adaptés à la saisie des différents champs de ces classes, en particulier :

- Des *EditText*, ou similaire, pour la saisie des informations textuelles ou numériques ;
- Des *Spinner* pour le choix d'une valeur parmi une liste (nationalité et secteur d'activité d'un employé) ;
- Des *RadioButton* pour le choix du type de la *Person*, un *Worker* ou un *Student* ;
- La saisie de la date d'anniversaire se fera à l'aide d'un *DatePicker*, un dialogue qui s'ouvrira lors de l'appui sur un bouton à côté du champ de saisie ou sur le champ de saisie (qui ne doit pas pouvoir être directement édité par l'utilisateur à l'aide du clavier). Si la date d'anniversaire n'est pas encore définie le *DatePicker* sera présélectionné avec la date du jour, si elle est définie le *DatePicker* sera présélectionné avec celle-ci.

Votre interface devra s'adapter à la saisie d'un type de *Person* ou de l'autre en fonction du choix de l'utilisateur, les zones de saisie dédiées à des informations concernant uniquement un étudiant ne devront pas être affichées lors de l'édition d'un employé, et vice-versa. Également vous mettrez deux boutons à disposition permettant de : 1) vider le formulaire, 2) valider le formulaire.

3. Le contrôleur

Il s'agit de l'activité qui accueillera la vue et implémentera les différentes fonctionnalités devant être offertes :

- L'initialisation de la vue en fonction d'un objet existant du modèle, par exemple *exampleWorker* ou *exampleStudent* au lancement de l'activité ;
- La création d'une instance d'un *Student* ou d'un *Worker* lors de l'appui sur le bouton de validation du formulaire avec les données saisies dans celui-ci puis l'affichage de l'instance créée dans les logs ;
- Supprimer le contenu de tous les éléments de la GUI lors de l'appui sur le bouton correspondant.

4. Questions complémentaires

Veuillez répondre aux 5 questions suivantes. Pour chacune d'entre elles, vous développerez votre réponse et l'illustrerez par des extraits de code.

- 4.1 Pour le champ *remark*, destiné à accueillir un texte pouvant être plus long qu'une seule ligne, quelle configuration particulière faut-il faire dans le fichier XML pour que son comportement soit correct ? Nous pensons notamment à la possibilité de faire des retours à la ligne, d'activer le correcteur orthographique et de permettre au champ de prendre la taille nécessaire.
- 4.2 Pour afficher la date sélectionnée via le *DatePicker* nous pouvons utiliser un *DateFormat* permettant par exemple d'afficher *12 juin 1996* à partir d'une instance de *Date*. Le formatage des dates peut être relativement différent en fonction des langues, la traduction des mois par exemple, mais également des habitudes régionales différentes : la même date en anglais britannique serait *12th June 1996* et en anglais américain *June 12, 1996*. Comment peut-on gérer cela au mieux ?
- 4.3 Veuillez choisir une question en fonction de votre choix d'implémentation :
- a. Si vous avez utilisé le *DatePickerDialog*¹ du SDK. En cas de rotation de l'écran du smartphone lorsque le dialogue est ouvert, une exception *android.view.WindowLeaked* sera présente dans les logs, à quoi est-elle due et comment pouvons-nous la corriger ?
 - b. Si vous avez utilisé le *MaterialDatePicker*² de la librairie *Material*. Est-il possible de limiter les dates sélectionnables dans le dialogue ? Ceci en particulier pour une date de naissance car il est peu probable d'avoir une personne née il y a plus de 110 ans ou à une date dans le futur. Comment pouvons-nous mettre cela en place ?
- 4.4 Lors du remplissage des champs textuels, vous pouvez constater que le bouton « suivant » présent sur le clavier virtuel permet de sauter automatiquement au prochain champ à saisir, cf. Fig. 2. Est-ce possible de spécifier son propre ordre de remplissage du questionnaire ? Arrivé sur le dernier champ, est-il possible de faire en sorte que ce bouton soit lié au bouton de validation du questionnaire ?
- Hint : Le champ remark, multilignes, peut provoquer des effets de bords en fonction du clavier virtuel utilisé sur votre smartphone. Vous pouvez l'échanger avec le champ e-mail pour faciliter vos recherches concernant la réponse à cette question.*

¹ <https://developer.android.com/reference/android/app/DatePickerDialog>

² <https://github.com/material-components/material-components-android/blob/master/docs/components/DatePicker.md>

4.5 Pour les deux *Spinners* (nationalité et secteur d'activité), comment peut-on faire en sorte que le premier choix corresponde au choix *null*, affichant par exemple le label « Sélectionner » ? Comment peut-on gérer cette valeur pour ne pas qu'elle soit confondue avec une réponse ?

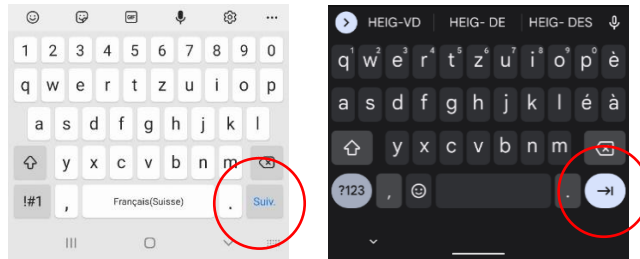


Figure 2 - Claviers virtuels (Samsung à gauche, et Google à droite) lors du remplissage de champs textuels. Les cercles rouges identifient un bouton permettant de passer automatiquement au prochain champ sans devoir fermer le clavier.

Durée / Evaluation

- 4 périodes, à rendre le dimanche **02.11.2025 à 23h59** au plus tard.
- Pour le rendu de votre code, nous vous demandons de bien vouloir zipper votre projet Android Studio en veillant à bien supprimer les dossiers *build* (à la racine et dans *app/*) pour limiter la taille du rendu. Vous remettrez également un document **pdf** comportant les explications sur l'implémentation de votre solution ainsi que les réponses aux questions posées.
- Merci de rendre votre travail sur *CyberLearn* dans un zip unique. N'oubliez pas d'indiquer vos noms dans le code, sur vos réponses et de commenter vos solutions.