



Devoir – Créer une fiche « Profil développeur » avec Compose

Contexte

Vous avez commencé à travailler avec Jetpack Compose et vous avez déjà manipulé plusieurs notions importantes : *les fonctions composables ; les paramètres ; les callbacks ; l'arbre de composition ; Modifier ; le chaînage des modifieurs ; la différence entre padding externe et padding interne ; la création de composants réutilisables ; les previews ; les premières notions d'accessibilité.*

L'objectif de ce devoir est de vérifier que vous êtes capables de réutiliser ces notions en autonomie, sans suivre un TP guidé.

Vous allez créer un petit écran **Compose** représentant une fiche « **Profil développeur** ». L'objectif est de produire une interface propre, structurée, lisible et composée de plusieurs composants réutilisables.

Sujet

Vous devez créer un écran Jetpack Compose affichant le profil d'un développeur ou d'une développeuse. Le profil peut être fictif ou inspiré de vous-même.

Votre écran devra contenir au minimum :

- un **nom** ;
- un **rôle** ou **titre** professionnel ;
- une courte **description** ;
- une **liste de compétences** ;
- une information de **contact** ou de **disponibilité** ;
- au moins une **action utilisateur** sous forme de **bouton** ou de **zone cliquable**.

Pour vous inspirer, voici des exemples de profils possibles : *développeur Android ; développeuse full-stack ; étudiant en informatique ; développeur backend ; développeuse mobile ; profil freelance ; profil junior en recherche de stage ou d'alternance ; profil fictif d'un personnage technique.*

Vous êtes libres sur le contenu, mais l'écran doit rester professionnel et lisible.

Cadre technique à respecter

Ce devoir doit rester dans le périmètre des notions déjà vues en cours.

Vous pouvez utiliser :

Composants Compose	Modifieurs et accessibilité	Structure Android / outils
@Composable Text Button Surface Column	Modifieur padding fillMaxWidth clickable semantics contentDescription Role.Button	MainActivity setContent thème généré du projet Scaffold innerPadding @Preview

Vous ne devez pas utiliser de notions qui n'ont pas encore été étudiées dans le cours.

Sont donc **interdits pour ce devoir** :

les appels API ;

remember ;

LazyRow ;



Retrofit;	mutableStateOf;	les grilles;
Room;	rememberSaveable;	les formulaires interactifs avancés;
ViewModel;	Navigation Compose;	la persistance de données;
StateFlow;	LazyColumn;	l'architecture MVVM complète.

Le clic sur un bouton ou une zone cliquable peut simplement appeler une **fonction vide** ou contenir un commentaire. Il n'est pas demandé de modifier réellement l'interface après le clic.

Travail demandé

Vous devez créer une application Android Compose simple contenant un écran principal de profil développeur.

Votre code doit être organisé en plusieurs composables.

Vous devez créer au minimum *5 fonctions composables*, par exemple :

- un composable pour **l'écran principal** ;
- un composable pour la **fiche de profil** ;
- un composable pour **l'en-tête du profil** ;
- un composable pour **une compétence** ;
- un composable pour **une action** ;
- un composable pour **une information secondaire**.

Ces noms sont donnés à titre **indicatif**. Vous pouvez choisir vos propres noms, à condition qu'ils soient clairs.

Contraintes obligatoires

Votre devoir doit respecter les contraintes suivantes.

1. Composants réutilisables

Au moins 4 de vos composables doivent recevoir des paramètres.

Exemples de paramètres possibles : *un nom ; un rôle ; une description ; une compétence ; une disponibilité ; un texte de bouton ; un callback de clic.*

L'objectif est d'éviter les composables trop figés qui affichent toujours les mêmes données.

2. Utilisation de Modifier

Chaque composable réutilisable doit accepter un paramètre : **modifier: Modifier = Modifier**.

Ce **modifier** doit être utilisé dans le composable.

L'objectif est de montrer que **vous savez laisser le parent décider** du placement externe du composant.

3. Padding externe et padding interne

Votre écran doit montrer clairement la différence entre :

- un **padding externe**, appliqué par le parent pour placer un composant dans l'écran ;
- un **padding interne**, appliqué dans un composant pour espacer son contenu.



Vous devrez être capables d'expliquer cette différence dans le **court commentaire** demandé à la fin du devoir.

4. Chaînage de modifieurs

Vous devez utiliser au moins **une chaîne de modifieurs** contenant plusieurs appels.

Exemple attendu dans l'esprit : *largeur ; padding ; clic ; sémantique ; ou autre combinaison pertinente.*

Il ne s'agit pas d'ajouter des **modifieurs** au hasard. La **chaîne** doit avoir un sens dans votre interface.

5. Action utilisateur

Votre interface doit contenir au moins **une action utilisateur**, par exemple :

- bouton « *Contacter* » ;
- bouton « *Voir le portfolio* » ;
- zone cliquable « *Disponible pour un stage* » ;
- bouton « *Afficher les projets* » ;
- bouton « *Envoyer un message* ».

Le **clic** n'a pas besoin de modifier l'interface. Il peut simplement **appeler un callback vide** ou une **fonction contenant un commentaire**.

L'important est de respecter le principe vu en cours : le **composant signale l'événement**, mais ne décide pas d'une logique applicative avancée.

6. Accessibilité de base

Si vous créez une **zone personnalisée cliquable**, vous devez lui associer une **intention** compréhensible (accessibilité). Par exemple :

- rôle de bouton ;
- description claire de l'action ;
- indication compréhensible pour un lecteur d'écran.

Vous devez au moins utiliser une information d'accessibilité sur un composant interactif personnalisé.

Si vous utilisez uniquement un **Button**, expliquez pourquoi ce composant est déjà plus explicite qu'une simple zone cliquable personnalisée.

7. Previews

Vous devez créer au moins deux **previews**.

Les deux **previews** doivent montrer **deux variantes différentes** du profil. Par exemples :

- un profil Android junior ;
- un profil full-stack ;
- un profil disponible ;
- un profil non disponible ;
- un profil avec des compétences différentes.

L'objectif est de vérifier que vos composables peuvent afficher **plusieurs jeux de données**.



Livrables attendus

Vous devez rendre un **dépôt Git public** contenant votre projet Android.

Le dépôt doit contenir :

1. Le projet Android complet.
2. Un fichier **README.md**.
3. Le code Compose de votre fiche « **Profil développeur** ».
4. Au moins deux **previews** fonctionnelles.
5. Une courte **réponse** écrite dans le **README.md**.

Contenu attendu du README.md

Votre fichier README.md doit contenir :

1. **Titre du projet** – Exemple - « Profil développeur Compose »
2. **Description courte** – Expliquez en 3 à 5 lignes ce que fait votre écran.
3. **Notions utilisées** – Listez les notions Compose utilisées dans votre projet. Par exemple :
 - composables ;
 - paramètres ;
 - modifieurs ;
 - padding externe / interne ;
 - callbacks ;
 - previews ;
 - accessibilité de base.
4. **Explication courte** – Répondez brièvement aux questions suivantes :
 1. Quels **composables** avez-vous **créés** ?
 2. Quel composable représente **l'écran principal** ?
 3. Où utilisez-vous un **padding externe** ?
 4. Où utilisez-vous un **padding interne** ?
 5. Quel composant reçoit un **callback** ?
 6. Quelle information **d'accessibilité** avez-vous ajoutée ?
 7. **Pourquoi** vos composants peuvent-ils être considérés comme **réutilisables** ?
5. **Lien ou capture** – Ajoutez au moins une capture d'écran de votre **preview** ou de **l'application lancée**.

Modalités de rendu

Le rendu se fait via Microsoft Teams.

Vous devez déposer :

- le **lien** vers votre **dépôt Git public** ;
- éventuellement une **capture d'écran** si non présente dans Git ;
- **aucun fichier ZIP** n'est demandé : déjà - c'est trop volumineux, et pas de votre niveau !

Le dépôt doit être accessible publiquement au moment de la correction.



Critères d'évaluation

Le devoir sera évalué selon les critères suivants :

- **Projet fonctionnel** : le projet s'ouvre dans Android Studio, compile et affiche un écran Compose.
- **Découpage en composables** : l'interface est organisée en plusieurs composables clairs, lisibles et bien nommés.
- **Réutilisabilité** : les composables utilisent des paramètres et acceptent un **modifier** quand c'est pertinent.
- **Utilisation de Modifier** : les **modifiers** sont utilisés correctement, avec un **chaînage** cohérent et une distinction visible entre **padding externe** et **interne**.
- **Interaction** : au moins une **action** utilisateur est présente et exposée via un **callback**, sans logique applicative avancée.
- **Accessibilité de base** : les éléments cliquables personnalisés sont compréhensibles pour l'utilisateur et les outils **d'accessibilité**.
- **Previews** : au moins deux **previews** montrent des variantes différentes de l'interface.
- **README** : le dépôt contient une description du projet, les réponses demandées, une capture et un lien Git exploitable.

Conseils

Commencez simple ! Ne cherchez pas à créer une interface trop complexe. Il vaut mieux rendre une fiche claire, propre et bien structurée qu'un écran visuellement chargé mais mal organisé.

Pensez à nommer vos composables clairement.

Évitez de copier-coller plusieurs fois le même bloc de code. Si une partie se répète, demandez-vous si elle peut devenir un composable réutilisable.

N'utilisez pas de notions non vues en cours. Ce devoir évalue votre capacité à maîtriser les bases Compose vues jusqu'à présent, pas à chercher des solutions avancées sur Internet ou via les IA.

Avant de rendre, vérifiez que :

- le projet **compile** ;
- les **previews** fonctionnent ;
- le **dépôt Git est public** ;
- le **README** est présent ;
- le **lien** déposé sur Teams **fonctionne**.