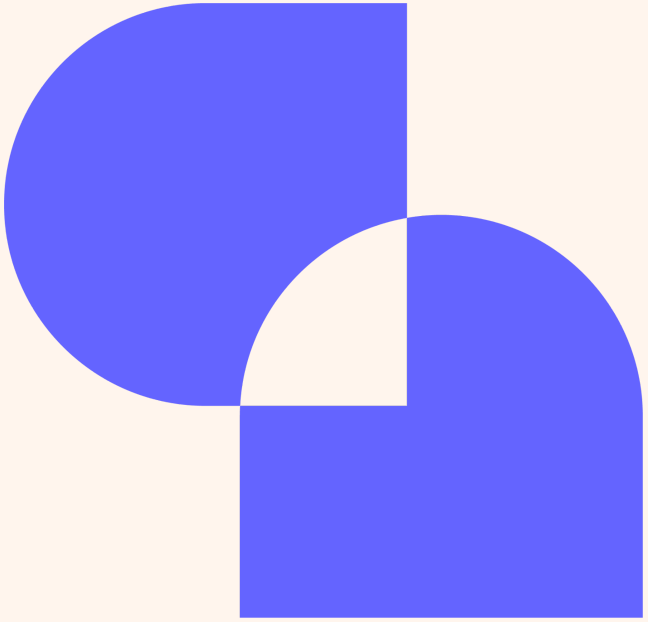
GoMind

“Agir avec l’esprit éveillé”

DOSSIER DE COMPÉTENCES

Développeur Sénior Kotlin

Spécialité : Compose/MVVM



PRINCIPALES EXPÉRIENCES

KINOMAP : Développeur Kotlin/Compose (depuis 2023-2025)
GETCLOVERS : Développeur Kotlin/Compose (2023)
DECATHLON : Développeur Kotlin/Compose (2022- 2023)
HEROES JOBS : Développeur Kotlin/Hilt (2018-2022)

CONNAISSANCES TECHNIQUES

Langages : Java, Kotlin, C/C++, Objective-C

Librairies/framework :
Kotlin, Retrofit, Room, Coroutine, Koin, Retrofit, Compose, Data class, Live Data, Flow, Arrow, Exposed, Exceptions, Navigation, Flat map, Safe call Operator, Nulls, Volley, Dagger Hilt, Gradle, Actor, Material, Firebase, Material 3,

Architecture/Design Pattern : MVVM, MVP, MVI, MVC, Repository, Clean Architecture, Modularisation, Singleton, Observer, Use Case, Factory, Flow, Design system

Base de données : SQLlite, MySQL, NoSQL, PostgreSQL, Data store preferences, External Storage, Room, GraphQL (connaissance)

Bonnes pratiques : Junit, Expresso, Volley, Mockk, Lint, SOLID, Turbine, TDD, BDD, DDD (connaissance)

Outils : Git, Gitflow, Gitlab, GitHub, Bitrise, SonarQube, Spot et Spotless Supply, Crashlytics, Analytics, App Distribution, CrashAnalytics, LeakCanary, TNR, KT Lint,

Gestion de projet : Jira, Figma, Scrum

EXPÉRIENCES TECHNIQUES

KINOMAP : Développeur Kotlin (depuis 2023-2025)

Développement from scratch de nouvelles fonctionnalités sur l'application Kinomap (mobile, tablette, TV) et des versions embarqués

Workflow :

- Analyse du besoin
- Rédaction des spécifications techniques
- Suivi des wireframe des UX via Figma
- Gestion de projet en Scrum :
  - Cérémonies : daily, retro, démo, sprint planning, refragment
  - Sprint 2 semaines
  - Conception de ticket Jira entre dév :
    - Évaluation des tickets
    - Spécification des comportements à développer et des critères d'acceptances
- Équipe 7 développeurs
  - Pair programming
  - Code Review
- Utilisation de l'architecture MVVM :
  - Gestion de Flow uni directionnel :
    - Interactions utilisateurs exposées dans le view model
  - Mise en place de la clean architecture
  - Gestion de dépendances avec Dagger Hilt
  - Utilisation de Compose :
    - Utilisation de Navigation
- Développement Kotlin/Compose :
  - Utilisation de Material Design 3
  - Intégration d'API, de socket de services backend pour synchroniser les données en temps réel
  - Utilisation de Retrofit pour la récupération la donnée côté server
  - Utilisation de Coroutine pour la gestion des flow
  - Utilisation de Compose pour l'affichage
  - Utilisation de Firebase Remote Configuration
  - Utilisation de AppDistrib
  - Utilisation de Crashlytics
  - Utilisation des Data class
  - Gestion du View binding
  - Gestion des états d'interfaces utilisateurs grâce aux mutables state flow
  - Utilisation de Room
- Versioning avec Git
  - Suivi du gitflow
- Écriture des tests unitaires avec Junit, mock et turbine
- Écriture de tests UI avec Expresso
- Utilisation de GitlabCI
- Utilisation de SonarQube, Spot et Spotless Supply

Environnement technique :
MVVM, Data Source, Kotlin, Coroutine, Dagger Hilt, Retrofit, Compose, Material Design 3 Jira, Figma, GitlabCI, SonarQube, Spot, Spotless Supply, Git, Gitflow, Junit, Expresso

GETCLOVERS : Développeur Kotlin (2023)

Refonte complète (UX/UI, Dev) de l'interface utilisateur de l'application. Mise en place d'un design system des composants d'applications.

- Rédaction des spécifications techniques et fonctionnelles
- Détermination des besoins et des features désirées
- Gestion de projet en Scrum :
  - Cérémonies : daily, retro, démo, sprint planning, refragment
  - Sprint 2 semaines
- Équipe 3 dev :
  - Code Review
  - Écriture de documentation technique
- Maintien de la clean architecture MVVM :
  - Création from scratch de la couche métier
  - Utilisation des Repository, Use case, Singleton
  - Maintien du code
  - Intégration de l'injection de dépendance avec Dagger Hilt
  - Évolution des écrans XML
- Développement Kotlin/XML :
  - Utilisation de Coroutine et gestion des life cycles
  - Utilisation des Data class
  - Maintien des écrans en View Binding
  - Gestion des états d'interfaces utilisateurs grâce aux live data
- Versioning avec Github
  - Suivi du gitflow
- Écriture des tests unitaires avec Junit et Mockk
- Utilisation de Firebase Crashlytics
- Mise en place de Crash analytics, collection (backend), connection SSO (Google, Apple et Facebook)
- Mise en place de tests unitaires
- Bonne pratiques :
  - Écriture de documentation
  - Pair programming
  - Review de code

Environnement technique :
MVVM, Kotlin, XML, Coroutine, Dagger Hilt, Repository, Bitrise, Room, Jetpack compose, Green DAO, NFC, QR Code, Junit, Mockk, Github, Gitflow, Figma, Jira, Firebase Crashlytics, Crash analytics, collection

DECATHLON : Développeur Kotlin (2022-2023)

Innovation et développement de nouvelles fonctionnalités en respectant les normes de qualité du code et les procédures de test établies

Workflow :

- Analyse du besoin
- Rédaction des spécifications techniques
- Suivi des wireframe des UX via Figma
- Gestion de projet en Scrum :
  - Cérémonies : daily, retro, démo, sprint planning, refragment
  - Sprint 2 semaines
  - Conception de ticket Jira entre équipe :

        - Découpage des tickets par use case
        - Spécification des comportements à développer et des critères d'acceptances
- Équipe 5 dev
    - Pair programming
    - Code Review
- Utilisation de l'architecture MVVM :
    - Gestion de Flow uni directionnel :
        - Interactions utilisateurs exposées dans le view model
    - Utilisation de SOLID
    - Mise en place de la clean architecture
    - Gestion de dépendances avec Dagger Hilt
    - Utilisation de Compose :
        - Utilisation de Navigation
- Développement Kotlin/Compose :
    - Utilisation de Material Design 3
    - Intégration d'API et de services backend pour synchroniser les données en temps réel
    - Utilisation de Retrofit pour la récupération la donnée côté server
    - Utilisation de Coroutine pour la gestion des flow
    - Utilisation de Compose pour l'affichage
    - Utilisation de Firebase Remote Configuration
    - Utilisation de Crashlytics
    - Utilisation des Data class
    - Gestion des états d'interfaces utilisateurs grâce aux mutables state flow
    - Utilisation de Room
    - Utilisation de LeakCanary (fuite de mémoire)
- Versioning avec Git
    - Suivi du gitflow
- Écriture des tests unitaires avec Junit
- Écriture de tests UI avec Expresso
- Pratique des tests TNR (non régression)
- Utilisation de GitlabCI,
- Utilisation de Sonar & Bitrise

Environnement technique :
MVVM, SOLID, Singleton, Repository, Data Source, Kotlin, Coroutine, Dagger Hilt, Retrofit, Compose, Material Design 3 Jira, Figma, GitlabCI Sonar, Bitrise, Room, Git, Gitflow, GitlabCI, Junit, Expresso, LeakCanary, TNR, Sonar, Bitrise

---

HEROES JOBS : Développeur Kotlin (2018-2022)

Responsable du développement et de l'évolution de deux applications Android en production, alliant correction de bugs, intégration de nouvelles fonctionnalités et optimisation des performances.

- Analyse du besoin
- Utilisation de spécifications techniques
- Gestion des tickets avec Monday
- Mise en place de briefing avec les équipes produits et Design pour le cadrage du projet
    - Participation à la priorisation des features
- Gestion de projet en Scrum :
    - Cérémonies : daily, retro, démo, sprint planning, refragment
- Équipe 6 dev :
    - Pair programming
    - Code Review
    - Écriture de documentation technique
- Utilisation de l'architecture MVVM :
    - Implémentation de SOLID

    - Utilisation de Singleton
- Développement Kotlin :
    - Mise en place de kotlin
    - Mise en place d'un design system
    - Utilisation des XML
    - Intégration des Coroutines
    - Utilisation des live data/RXJavaMise en place des analyses via Firebase
    - Utilisation de CrashAnalytics
    - Mise en place des analytics
    - Gestion et publication des applications sur le Playstore
    - Intégration d'API et de services backend pour synchroniser les données en temps réel
    - Utilisation de Retrofit pour la récupération la donnée côté server
    - Utilisation de Room
    - Gestion des push notifications
- Versioning avec Git
- Écriture des tests unitaires avec Junit
- Écriture de Test UI Expresso
- Utilisation de KT Lint
- Mise en place de la CI avec GitlabCI
- Bonne pratiques :
    - Écriture de documentation
    - Pair programming

Environnement technique :
Kotlin, XML, Coroutines, RxJava, Volley, Junit, Expresso, KT Lint, GitlabCI, Firebase, CrashAnalytics, Retrofit, Room

LANGUES

Anglais

Français