Document explicatif du projet technique

Frezy VAMBE

Durée de réalisation du projet

Dans le soucis de rendre un projet de qualité mettant en avant mes compétences professionnels en développement mobile iOS, cela m'a pris au trop 6 jours pour réaliser ce projet. Ce projet a été fait avec Xcode 11.3.1 (11C504) pour tous les Devices sous iOS 13 et macOS 10.15

Architecture du projet

Ce projet modulaire est composée d'un Workspace qui contient 4 sous-projets qui sont:

- **Wews iOS :** L'application mère iOS Catalyst (iPad, iPhone, Mac)
- **Core:** Ce projet contient tout le coeur métier de l'application mère. Il s'agit notamment des modèles et services (WEB Services dans ce cas) consommés par l'application mère.
- **UIComponentKit:** Ce projet contient des composants graphiques réutilisables permettant d'accélérer le développement des écrans de l'application mère.
- **NetworkKit:** C'est une couche réseau lightweight et réutilisable pour faire des appels réseaux au sein de l'application.

Pour ce qui est du **patron architectural** utilisé au sein de l'application mère, je suis parti sur du MVVM avec du Combine qui est la nouvelle framework d'Apple pour faire de la programmation fonctionnelle réactive. Combine tout comme RxSwift fournit des flux logiques de données qui peuvent émettre des valeurs et éventuellement se terminer par un succès ou une erreur.

Choix technique

Dans l'application mère, le patron de conception utilisé pour la navigation est le **Flow Coordinator**. Ce design pattern de navigation permet d'éviter d'avoir des Massives ViewControllers en déplaçant tous les appels UIKit spécifiques à la navigation dans des objets (Coordinators) crées à cet effet.

Pour l'**injection de dépendance** au sein du projet, j'aurais pu utiliser Swinject dans un cadre beaucoup plus professionnelle et sur une grosse application. Cependant, comme cette application est assez simple, j'ai fais l'injection de dépendance manuellement.

Je n'ai pas expressément mis de linter tels que **Swiftlint**. Swiftlint est très utile car il permet de garder une base de code cohérente et maintenable dans un projet avec une équipe de développeurs. Cela peut parfois être très difficile dans ce genre de projets car chaque dévelopeur a des conventions et des styles de programmation différents. De plus, les différents niveaux d'expérience avec le langage d'un développeur à un autre peuvent entraîner la plupart du temps une application très difficile à déboguer et généralement très difficile à comprendre pour un nouveau développeur qui rejoint l'équipe. Etant le seul développeur au sein de ce projet, je n'ai pas trouvé nécessaire de le mettre en place.

J'aurais pu utiliser les fichiers de configurations d'Xcode pour gérer les différents environnement(**Development, Production, Beta**) et registre de build au sein de cette application mais en l'implémentant au sein de ce projet, j'ai rencontré des soucis de configuration au niveau des schèmes. J'ai pu identifier que ces soucis étaient liés à la modularité du projet et à la configuration des different schemes avec les fichiers de

configuration d'Xcode. Pour cela et afin de ne pas passer trop de temps dessus, j'ai préféré l'enlever.

Je n'ai pas mis en place d'intégration continue par manque de temps. Le projet en entier contient actuellement **38 tests unitaires**.

L'application est disponible sur **iPhone, iPad et Mac (Catalyst)**. Cependant, je n'ai pas géré l'adaptation aux écrans pour la plateforme MacOS afin de ne pas passer plus de temps mais c'est quelque chose que j'aurais fait dans un cadre beaucoup plus professionnel.