Choix tech mobile

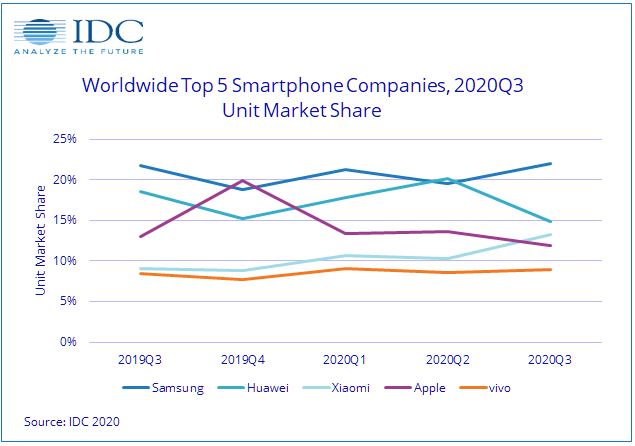
## **Choix Technologie pour le site Mobile**

Dans l’écosystème mobile, deux acteurs se taillent la part du Apple (avec iOs) et Google (avec Android). D’autres plateformes existes (RIM avec BlackBerry ou Windows avec Windows Phone), mais ils restent marginales

Lorsque l’on fait du développement pour mobile, il convient de bien choisir son périmètre ( un seul os, les eux, ..), de définir le type de développement(natif ou multiplateformes) et de bien entendu le choix de l’outil de dévelopement.

# Le périmètre

Selon IDC en 2020, Samsung et Apple continuent à être en tête du marché, même si ils sont talonnés de près par Huawei et Xiaomi. Xiaomi étant avec un OS Android et la plupart des Huawei aussi -,seul les plus récents implémentent le nouvel OS de Huawei.





Source IDC : <https://www.idc.com/getdoc.jsp?containerId=prUS46974920>

Il paraît donc judicieux au vu de ces chiffres de développer à la fois pour Android et pour IOS.

# Développement natif ou multiplateformes

Le développement d’un application native se fera pour un système d'exploitation précis (Android, iOS, Windows Phone,) avec des technologies qui lui est propre, par ex. pour iPhone, nous utiliserons le langage Objective-C ou Swift, pour Android, ce sera Java ou Kotlin, etc.

L’avantage de ce type de développement est qu’il permet d’exploiter l'intégralité des fonctionnalités natives des appareils (GPS, appareil photo, etc.), mais cela oblige à développer deux fois l’application, au risque d’avoir des différences entre les deux développements et de doubler le temps.

Le développement multiplateforme fait appel à des outils qui permettent de générer un code qui peut être déployé sur plusieurs OS. Ce type de développement présente de nombreux avantages :

* Nous avons 1 seul code et donc uniformité du code et, de la logique, mais aussi de l’apparence et de l’expérience utilisateur.
* Toutes les mises à jour et correctifs seront également automatiquement reflétés partout.
* Nous avons une économie de ressource car une seule équipe est nécessaire pour travailler sur le projet.
* La portée du code développé est plus importante car il peut être déployé sur de multiple plateforme et donc touché plus de cibles.

Le développement multiplateforme peut néanmoins avoir des inconvénients :

* Les outils de développement multiplateformes ne disposent pas de toutes les fonctionnalités offertes par chaque plate-forme, cela peut conduire à la mise en place de solutions contournement et donc du temps de développement en plus.
* Certaines fonctionnalités et intégrations natives ne sont pas disponibles dans les applications multi plateformes, ce qui limite l'expérience utilisateur que l’application peut fournir.

# Choix du framework

## Les différents frameworks

Lors des recherches effectués par l'équipe, 3 frameworks tiennent le haut du pavé

### XXXX

### XXXX

### XXXX

## Comment s'est fait notre choix ?

N'ayant pas forcement le temps de développer l'apprentissage des ces différents framework, nous nous sommes d'abord appuyer sur plusieurs critères

1. Sa pérennité

2. Sa popularité

3. La maitrise de l'outil dans l'équipe

# En conclusion

Nous avons choisi d'utiliser React car :

- Nous avons déjà des compétences dans ce framework dans l'équipe.

- Nous pouvons solliciter l'aide des professeurs sur ce framework.

- Il dispose d'une grande communauté et donc nous pourrons au besoin nous adresser à elle en cas de blocage.

- Sa popularité nous laisse à penser qu'il peut représenter un atout du point de vue professionnel.

https://www.synbioz.com/blog/tech/le-guide-ultime-du-developpement-mobile

https://fr.goodbarber.com/creer-app/48-developper-application-mobile-debutant/

<https://www.appbrain.com/search?q=react+native&src=stats&sort=relevancy>

ICI

# DEv native ou multiplateforme.

Incohérence avec les plateformes : Les outils de développement multiplateformes ne disposent pas de toutes les fonctionnalités offertes par chaque plate-forme => devrez peut-être utiliser des solutions de contournement. Cela peut également prendre un certain temps à ces cadres pour se mettre à jour avec les dernières fonctionnalités et mises à jour publiées par les plates-formes.

Fonctionnalités limitées : Il existe de nombreuses fonctionnalités et intégrations natives uniquement disponibles dans chaque plate-forme qui ne sont pas disponibles dans les applications multiplateformes, ce qui limite l'expérience utilisateur que vous pouvez fournir.

Tech multiplatform => évite de faire plusieurs dev

ce qui et de proposer des outils riches et rapides.

Mais si application utilisée sur Android et IOS => développée deux fois. Par ailleurs, elles risquent de présenter des différences, car développées par 2 équipes différentes.

**Une application native** = développée spécifiquement pour un système d’exploitation précis (Android, iOS, Windows Phone,) avec **des technologies qui lui est propre**.

EX iPhone => langage Objective-C ou Swift.

Android => ce sera Java ou Kotlin, etc.  
Avantage de ce type de développement et d’application => exploiter l'intégralité des fonctionnalités natives des appareils (GPS, appareil photo, etc.) et de proposer des outils riches et rapides.

Mais si application utilisée sur Android et IOS => développée deux fois. Par ailleurs, elles risquent de présenter des différences, car développées par 2 équipes différentes.

=> donc d’autres outils de développement (ou framework) sont apparus pour générer des applications compatibles avec plusieurs plateformes. Ils permettent un développement unique de l’application déployée sur les différents systèmes d’exploitation. Désormais, il n’est plus question de réfléchir au support visé, mais à son contenu !

développement multiplateforme?

Permet de créer des applications mobiles pour plusieurs plates-formes telles que iOS et Android avec une seule pile technologique.

=> 1 version de l’application => Pas plusieurs versions => mais 1 seul code déployé sur plusieurs plateforme

Avantages du développement multiplateforme

1 seul code => uniformité du code, de la logique

Uniformité de l’apparence, de l’expérience utilisateur

Toutes les mises à jour et correctifs sont également automatiquement reflétés partout.

Économiser des ressources => 1 seul équipe et pas plusieurs qui travaille sur les différentes versions natives. Cela vous permet d'exploiter des équipes plus petites et un temps de développement plus rapide pour gagner du temps et de l'argent.

Audience portée => La publication de votre application sur plusieurs plates-formes vous permet d'augmenter votre portée sur le marché sans effort supplémentaire

Inconvénients du développement multiplateforme

Les problèmes de performance : les dev multiplateformes, peuvent ne pas s'intègrer de manière transparente avec les plates-formes respectives et ont une communication incohérente entre les composants natifs et non natifs, réduisant la vitesse de l'application et dégradant les performances. .

Incohérence avec les plateformes : Les outils de développement multiplateformes ne disposent pas de toutes les fonctionnalités offertes par chaque plate-forme => devrez peut-être utiliser des solutions de contournement. Cela peut également prendre un certain temps à ces cadres pour se mettre à jour avec les dernières fonctionnalités et mises à jour publiées par les plates-formes.

Fonctionnalités limitées : Il existe de nombreuses fonctionnalités et intégrations natives uniquement disponibles dans chaque plate-forme qui ne sont pas disponibles dans les applications multiplateformes, ce qui limite l'expérience utilisateur que vous pouvez fournir.

Tech multiplatform => évite de faire plusieurs dev

Plusieurs techno possible

<https://www.npmtrends.com/react-native>

<https://blog.logrocket.com/flutter-vs-react-native-vs-xamarin/>

<https://www.appbrain.com/stats>

<https://www.npmtrends.com/react-native>

<https://insights.stackoverflow.com/survey/2020>

<https://www.ideematic.com/actualites/2020/08/flutter-le-prochain-leader-du-developpement-dapplication-multiplateforme%E2%80%89/>

https://www.maximelb.com/quel-framework-pour-developper-une-application-mobile-en-2020/

https://blog.logrocket.com/flutter-vs-react-native-vs-xamarin/

Recherche