



INF 3511 Programmation des Mobiles: Environnements Mobiles


Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

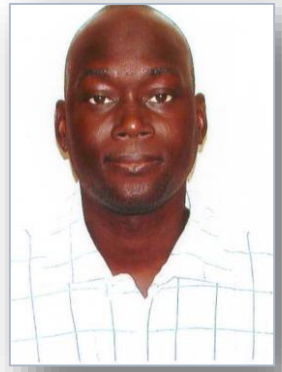
Université de THIES - UFR Sciences et Technologies -Département Informatique

Partie 3 :
Approches de développement mobile



Approches de développement mobile

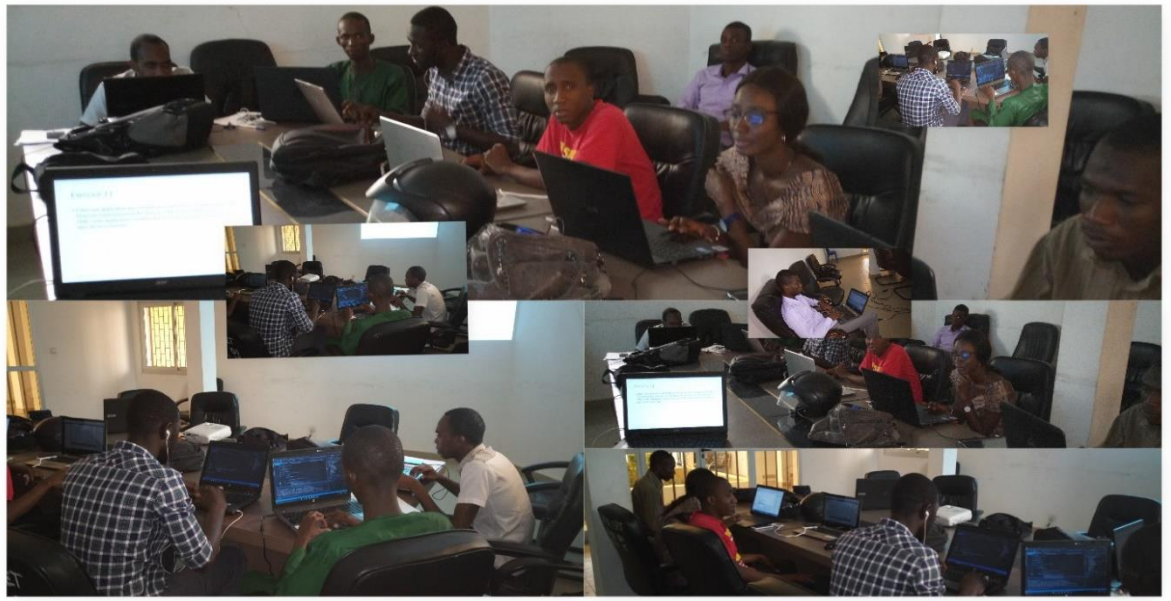
A propos de moi



- Enseignant-Chercheur à l'UFR SET- Université de THIES http://sites.univ-thies.sn/osall751/
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - osall@univ-thies.sn
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.

Une sagesse chinoise…

« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »
[Proverbe chinois]



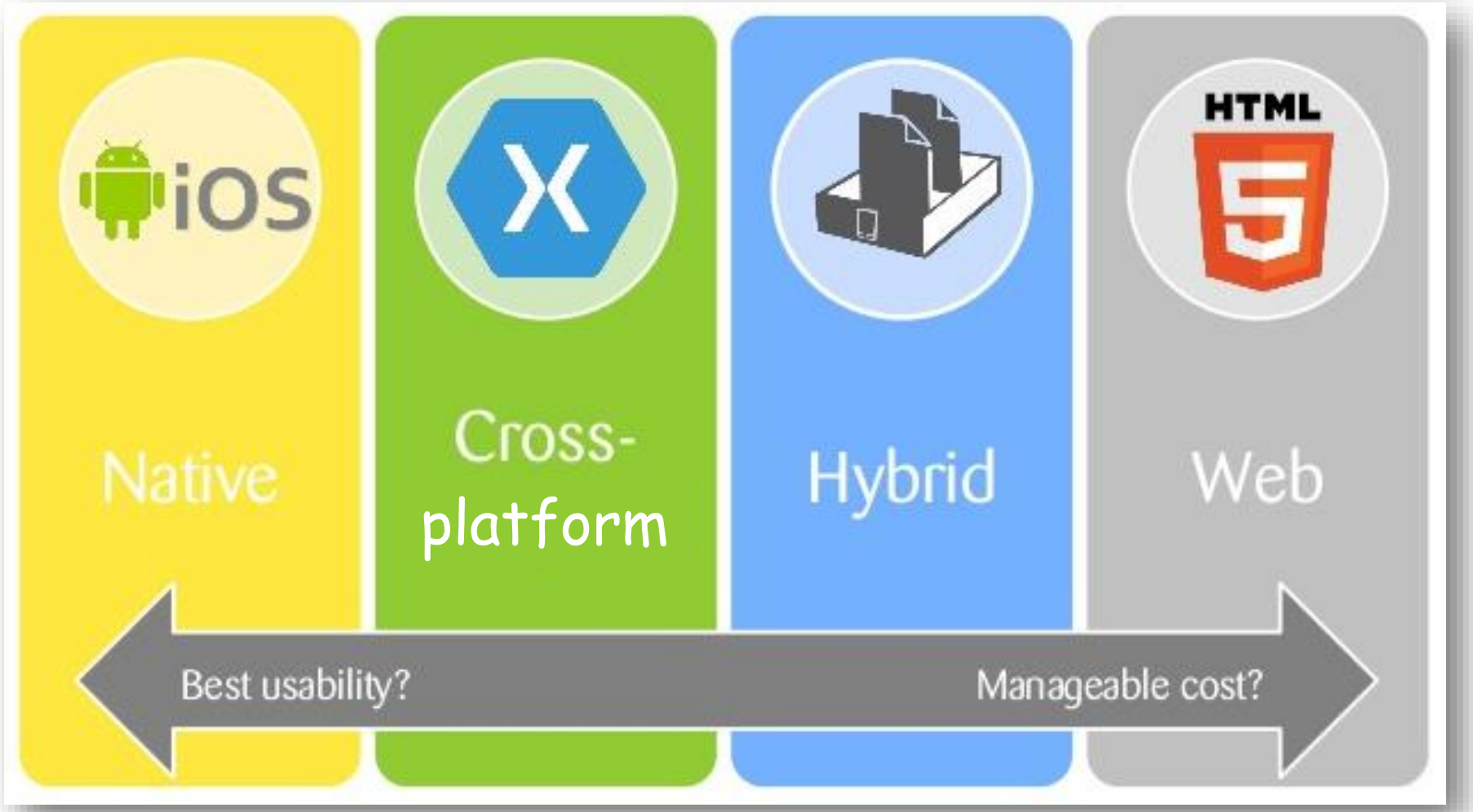
Contenu

1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
3. **Approches de développement mobile**
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android

Différentes approches pour le Développement Mobile

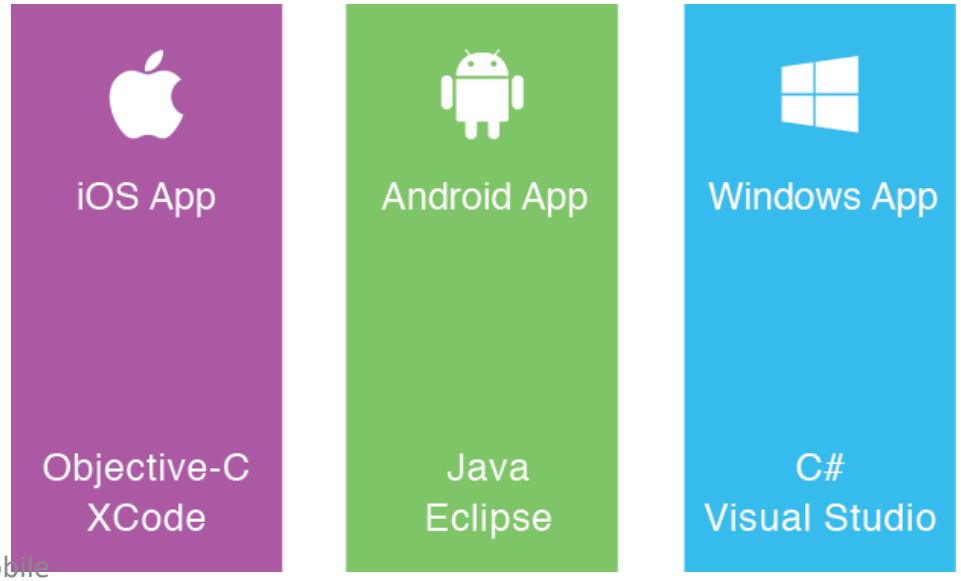
- **Plusieurs options pour développer une application mobile :**
  - **Développement natif** (Objective C ou swift, java, C#…)
  - **Solutions hybrides** : Des technologies de type Phonegap ou Cordova permettent d'embarquer une web view dans une application native. Ceux-ci permettent d'utiliser les fonctionnalités natives des smartphones. De plus elle pourra être distribuée en tant qu'application sur les plateformes d'applications (App Store, Android Market, etc.).
  - **Développement web:** utilisant le HTML5, CSS, JavaScript,…
  - **Solutions cross plateformes(natives)** : solutions comme Xamarin permettent de construire des applications iOS, Android et Windows mobile dans un environnement de développement Microsoft (C#)

Différentes approches pour le Développement Mobile



L’approche native: L’Approche en Silo

- **Les applications natives** sont spécifiques à une plate-forme mobile donnée (iOS ou Android) utilisant les outils de développement et le langage pris en charge par la plate-forme (par exemple Xcode et Objective-C avec iOS, Eclipse et Java avec Android). Les applications natives ont un design plus apprécié et fonctionnent plus performant.
- **Créer l’application plusieurs fois**
  - Plus de développeurs
  - Plus de maintenance
  - Différents environnements



L’approche native: L’Approche en Silo

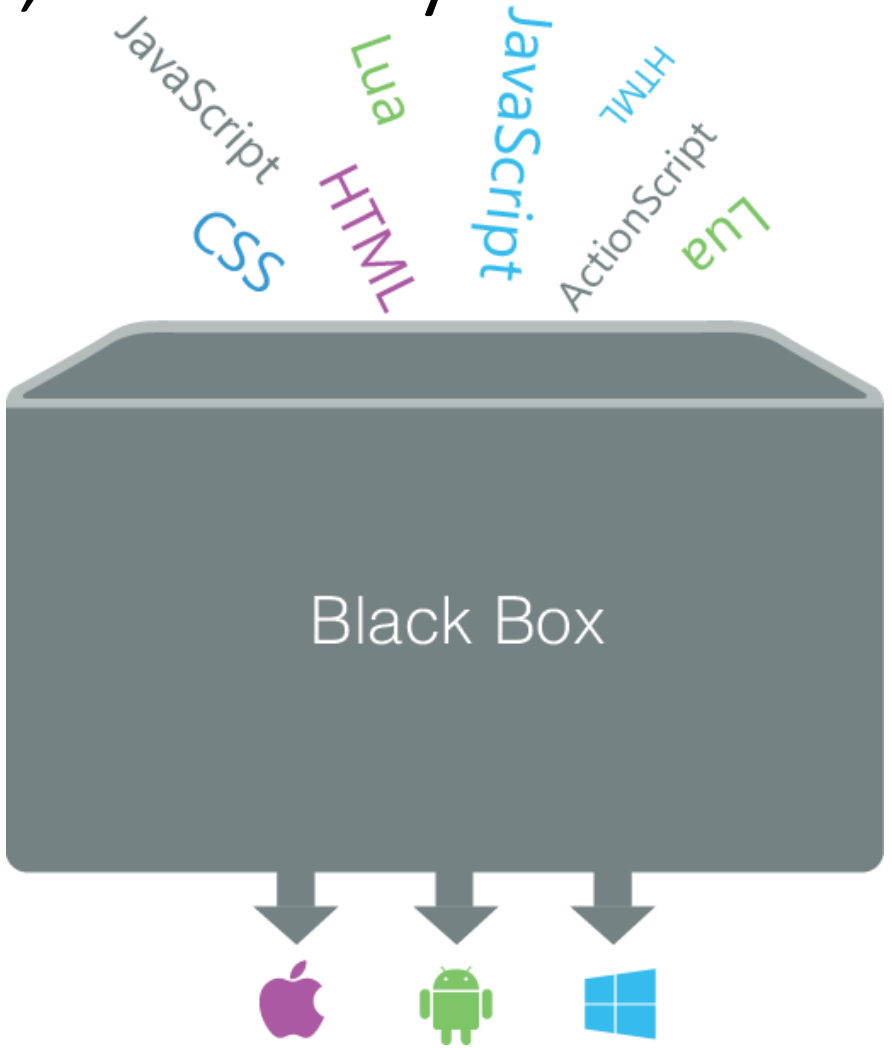
- **Avantages :**
  – Chaque système d’exploitation a sa version native – Possibilité de faire appel aux fonctionnalités de l’appareil utilisé (telle que la camera, la géolocalisation, le gyroscope …) et ainsi les exploiter de manière infinie… – Profiter d’une visibilité sur les stores d’applications et optimisation du référencement – Accès aux APIs natives – Plus simple à développer et plus d’outils à disposition et de support en cas de problème – Intégration à la facturation des stores si l’objectif est de rentabiliser le produit (Apps payantes) – Facilité à se conformer aux ergonomies spécifiques de chaque système d’exploitation afin d’avoir la meilleure expérience utilisateur possible.

L’approche native: L’Approche en Silo

- **Inconvénients :**
  – Le cout de développement est important car il faut développer une version spécifique pour chaque système d’exploitation – Les performances sont très souvent meilleures en natif – Toutes les mises à jour nécessiteront un coût de développement supplémentaire multiplié par le nombre de plate-formes utilisées

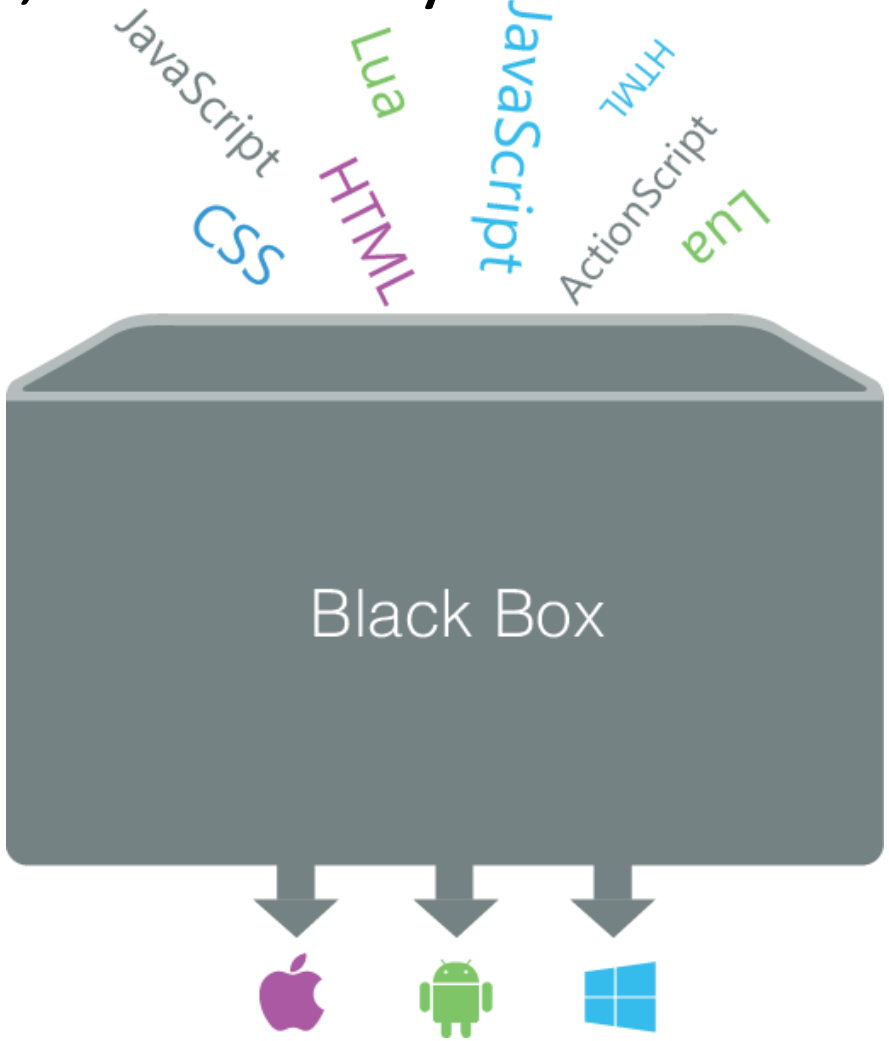
L’approche web: Write Once, Run Anywhere

- Utilise des technologies Web standard, généralement HTML5, JavaScript et CSS. Développée à grand renfort de codes javascript et autres boites à outils de type Angular ou React
- Approche de développement mobile permettant de créer des applications mobiles multiplateformes fonctionnant sur plusieurs appareils.



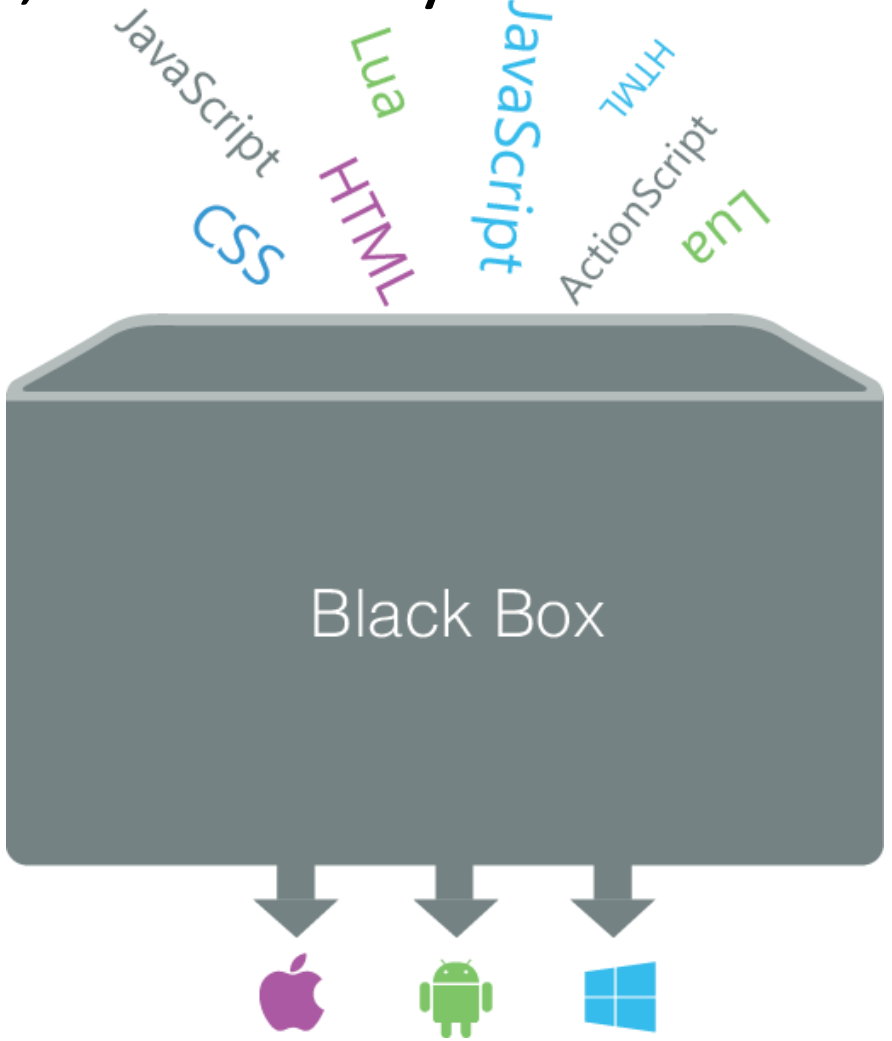
L’approche web: Write Once, Run Anywhere

- **Avantages :**
  – Développement web classique donc très bien maitrisé par les développeurs.
  – Utilisation de HTML5, CSS3, JavaScript et de frameworks – Responsive design pour un affichage optimal et adapté aux différents supports (smartphones, tablettes, desktop…) – Facilité de mise à jour du contenu



L’approche web: Write Once, Run Anywhere

- **Inconvénients :**
  – Nécessité d’avoir une connexion internet pour accéder au contenu – Pas de visibilité sur les stores d’application – Accessible via le navigateur. -Pas d’icone d’application – Ne s’exécute pas en plein écran



Application native vs. Application Web

- Développement spécifique selon le navigateur web
- Manque d'intégration avec l'OS(capteurs, partage, etc)
- Expérience utilisateur
- Performance « dégradées »

Facebook Web

Facebook natif

Approche hybride

Application utilisant le navigateur web intégré du support (Smartphone ou tablette) et les technologies Web (HTML, CSS et Javascript) pour fonctionner sur différents OS (iOS, Android, Windows Phone, etc.).
Une telle application utilise les fonctionnalités natives des Smartphones et peut être distribuée sur les plateformes d'applications telles que l'AppStore, le Google Play, etc. https://fr.wikipedia.org/wiki/Application_hybride

- Une application hybride est comme une application Web, principalement construite à l'aide de HTML5 et de JavaScript, puis encapsulée dans un conteneur natif mince qui donne accès aux fonctionnalités de la plate-forme native.
- Il existe de nombreux frameworks mobiles hybrides tels que Ionic, NativeScript, React Native, Xamarin, PhoneGap etc. PhoneGap est un exemple de conteneur le plus populaire pour créer des applications mobiles hybrides.
- Le développement hybride combine le meilleur (ou le pire) des mondes natif et web.

Approche hybride

- Cependant la qualité, la performance, et la résolution de ces applications sont nettement inférieures à celles des applications natives.
- En effet l'application hybride peut ne pas bien s'adapter au système d'exploitation utilisé par le smartphone du mobinaute (interface polluée par des widgets inutiles, mauvaise résolution etc.).
- En plus les applications hybrides ne sont accessibles que sur iPhone et Android, et sont parfois refusées sur certaines plateformes d'applications.

Approches multiplateformes

The developers use the cross-platform mobile development solutions to develop the mobile application once and run it on many platforms.

- Il n’existe pas d’interopérabilité entre les langages utilisés pour le développement mobile ciblant les différentes plateformes.
- Créer des applications Android, iOS et Windows Phone - de la logique métier à l'interface utilisateur - avec un code presque 100% commun comme C# sous Xamarin qui sera généré une application en code natif de la plateforme cible.

Le marché des applications multiplateformes devrait atteindre 7,5 millions de dollars d'ici 2018, et le nombre d'outils de développement multiplateforme est en hausse. Source https://blog.octo.com/etat-de-lart-des-solutions-cross-platform-mobile/

Pour les catégories d’app cross-plateformes. Voir : http://www.sciencedirect.com/science/article/pii/S2090447915001276

Quatre principales approches multiplateformes: Web, Hybrid, Interpreter and Cross-Compiler

- **Approche Web**: Une application créée avec l'approche Web est essentiellement un site Web accessible via le navigateur Web du téléphone. L'application est créée avec HTML, CSS et Javascript et est téléchargée pour fonctionner sur le smartphone.
- **Approche Hybride**: Hybride utilise le même concept que l'approche Web, mais n'exécute pas l'application dans un navigateur Web mais dans un conteneur natif sur le périphérique (fourni par le framework).

Quatre principales approches multiplateformes: Web, Hybrid, Interpreter and Cross-Compiler

- **Approche Interprétée(**Interpreter**)**: une application interprétée est téléchargée sur le périphérique des utilisateurs et l'interpréteur décide à l'exécution quel code doit être exécuté en fonction du périphérique en cours. L'approche interprétée utilise une couche d'abstraction pour permettre l'accès aux entités natives.
- **Approche Cross-Compiler**: Cross-Compiler prend du code et sort du code natif ou des binaires pour chaque plate-forme spécifique. Le code natif donné est ensuite compilé à nouveau sur les différentes plates-formes. L'application dépend donc de l'efficacité du Cross-Compiler.

Approche multiplateformes

- **Avantages :**
  – Chaque système d'exploitation a sa version native – Profiter d'une visibilité sur les stores d'applications – Les modifications et mises à jour seront effectives sur chaque plate-forme – Economie de budget – Idéale pour des applications de jeu en 2D ou en 3D
- **Inconvénients :**
  – Problème d'accès aux APIs – Problème de validation des applications par les stores – Difficile à maintenir et à faire évoluer – Toutes les fonctionnalités des appareils ne peuvent pas encore être exploitées – Problème d'ergonomie. UX et UI non optimisées

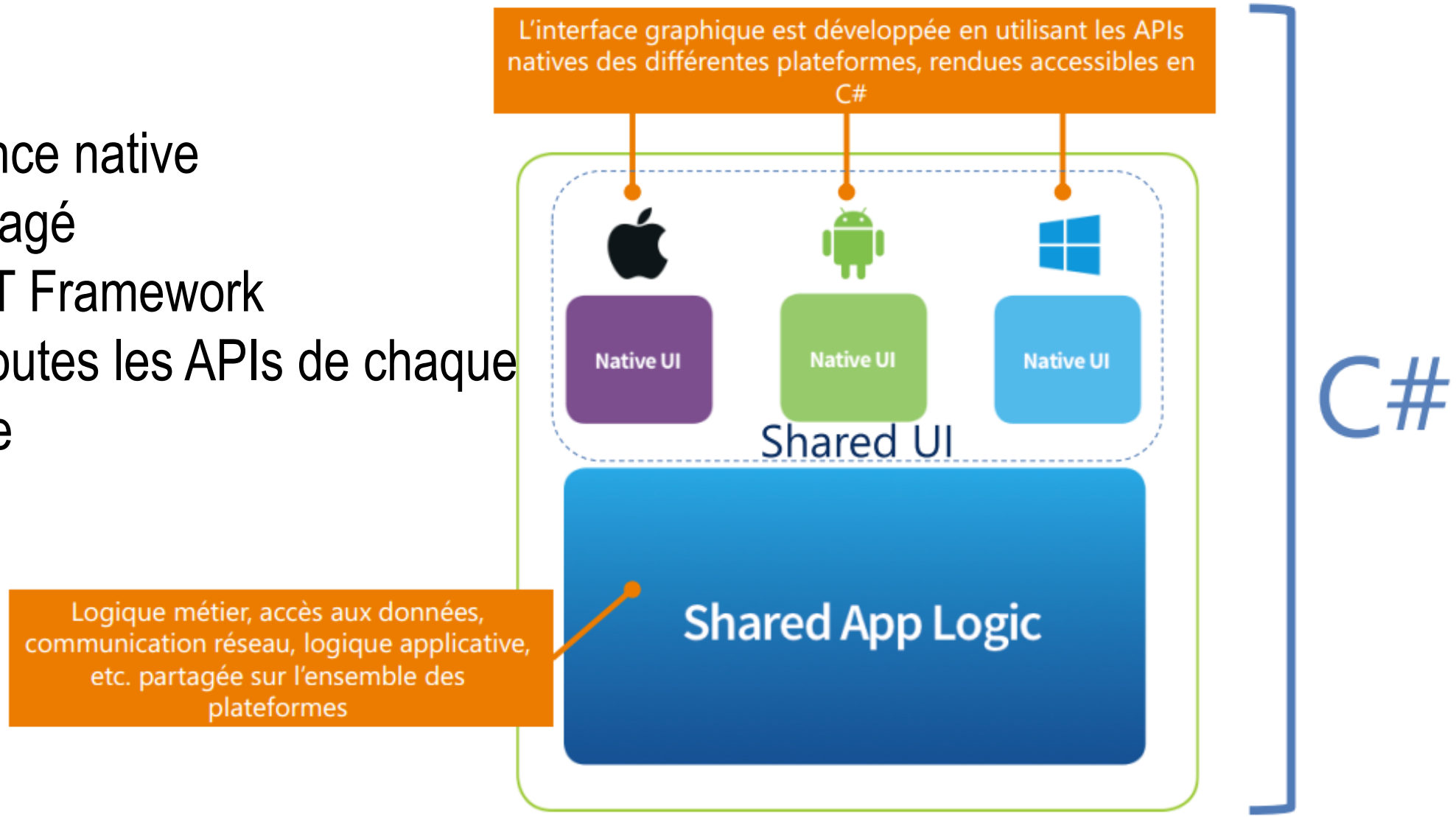
Xamarin

Xamarin, qui appartient à Microsoft, utilise son propre IDE pour le développement (Visual Studio sur Windows et Xamarin Studio sur OSx). La langue utilisée est C # et est conçue en utilisant leur IDE. Le code est compilé en code natif pour iOS et un fichier APK natif (Android Application Package). En 2017, Xamarin est au moins utilisé par 1,4 million de développeurs autour du monde et a d'autres applications dans sa suite pour les tests, etc.

- Xamarin est une société américaine fondée en mai 2011
- S’appuie sur le développement de la technologie mono
- Mono est créé en tant que projet open source, en vue de proposer une implémentation de la plate-forme .Net sous UNIX
- En février 2016, Xamarin est rachetée par Microsoft
- Mise à disposition gratuite de Xamarin dans Visual Studio Community Edition.

Approche multiplateformes: L'approche de Xamarin

- UI native
- Performance native
- Code partagé
- C# & .NET Framework
- Accès à toutes les APIs de chaque plateforme



Défis du développement d'applications mobiles

- Un paysage de fournisseurs concurrentiel et fluide (Apple, le consortium Android, notamment Amazon, RIM, HP) signifie que les applications doivent être multiplateformes pour une large adoption
- Pas d'appareil "standard" (qu'en est-il des appareils iOS, Windows Phone?)
- Faible entrée de bande passante (dans la plupart des cas, qu'en est-il des tablettes?)
- Taille d'écran limitée (tablettes?)
- Un manque de fiabilité de la connectivité et du périphérique (accès réseau, alimentation, lumière ambiante, bruit, au moins pour l'instant)
- Intégration des compromis avec les services cloud et d'entreprise

Supports du développement d'applications mobiles

- Langages orientés objet de troisième génération (iOS - Objective C ou Swift, Android - Java, Windows Phone - C #)
- Langages de script (JavaScript, Ruby)
- Structures multiplateformes - Titane, RhoMobile, Xamarin, PhoneGap
- Intégré dans des "frameworks" spécifiques pour le développement d'applications mobiles

Points clés

- iOS et Android ont participé à réduire la **fragmentation de l'industrie mobile** et ont attiré ainsi rapidement de nombreux développeurs
- iOS et Android sont à l'origine de l'**essor des smartphones et des marchés de l'Internet mobile**, tandis que les tentatives précédentes de type WAP et de « Walled gardens » ont échoué.
- Ce contexte a été favorable au développement d'**une économie dynamique des applications mobiles en Europe**, avec près de 1,5 million d'emplois et 13 milliards EUR de revenus générés (applications payantes, publicités) en 2016.

https://fr.idate.org/evolution-de-lecosysteme-mobile/

Points clés

- De nombreuses **success stories** ont été rendues possibles indépendamment de iOS et Android, notamment dans le domaine des jeux sur mobile (ex. Angry Bird de Rovio) ou dans l'écosystème de Facebook.
- **La concurrence entre les acteurs des terminaux a été renforcée**, plus intense qu'avant le lancement de iOS et Android. Android en particulier ayant notamment favorisé l'arrivée de nouveaux entrants grâce à un abaissement des prix.
- **Les développeurs utilisent couramment l'approche hybride ou multi-plateformes**, notamment en tirant profit des outils cross-platform.

https://fr.idate.org/evolution-de-lecosysteme-mobile/

Travaux dirigés

- Qu’est-ce qu’une approche de développement ?
- Lister les approches de développement mobile
- Décrire les approches de développement mobile
- Décrire brièvement chaque approche
- Listes 03 avantages et 03 inconvénients de chaque approche
- Expliquer l’approche multiplateforme et donner quelques raisons liées à son succès
- Lister 5 défis du développement mobile
- Quels sont les supports du développement d'applications mobiles ?

Travaux dirigés

- Quelles sont les motivations liées au développement Cross-Platform ?
- Qu’est-ce-que Xamarin?
- Qu’est qui a été à l’origine de l’essor des marchés de l’Internet mobile ?
- Citer une des applications mobiles de jeux qui connait un success-story actuellement au Sénégal
- Faire un résumé des points clés sur les plateformes mobiles

Webography

- https://ionicframework.com/
- https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3
- https://developer.xamarin.com/guides/
- https://angular.io/
- http://typescriptlang.org
- http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx
- Tutos Xamarin sur Google et Youtube entre autres.