UVS
UNIVERSITE VIRTUELLE DU SENEGAL

Les applications Hybrides /IONIC
Initiation au développement mobile

Ndiogou THIAO

PLAN

Sequence 4 : Les application Hybrides / Ionic

Introduction

- Bien que les deux géants que sont Google et Apple représentent une large majorité du marché des smartphones, tablettes, smart-tv et autres appareils connectés (système d'exploitation s'entend), ce dernier n'en est pas moins très fragmenté avec un certain nombre d'autres acteurs dont évidemment Windows Phone mais également BlackBerry OS, Firefox OS, Sailfish OS, Tizen…

- La tendance actuelle au "tout natif" n'est donc pas sans poser de sérieuses difficultés

- Une telle optique peut rapidement revenir à doubler voire tripler les ressources nécessaires à un projet et maintenance comprise, représente un investissement considérable tout au long de la durée de vie de l'application.

Les applications Hybrides (1/2)

- Observant le gain de notoriété des applications natives, et par conséquent, l'accroissement de la problématique du multiplateforme, des entreprises se sont rapidement rendues compte de l'importance de trouver des solutions permettant d'uniformiser le développement natif.

Les applications Hybrides (2/2)

- Des noms tels que PhoneGap, Appcelerator et Xamarin sont ainsi devenus aujourd'hui des incontournables du développement mobile multiplateforme grâce au succès de leurs solutions. Celles-ci sont notamment présentées plus en détails par la suite et sont utilisées ou citées dans le chapitre suivant consacré au développement d'une application prototype.

- Le choix de ces solutions a été fait principalement en raison de leur popularité et des différentes approches qu'elles représentent, il est toutefois important de noter que ce travail traite d'un domaine en constante progression dont le nombre d'acteurs a fortement augmenté ces dernières années, il est donc tout à fait possible que les rapports de popularité aient évolué lors de la lecture de ce document.

Le concept générale (1/3)

Explication du terme “Hybide”

- Partant du constat que l’utilisation d’un duo environnement de développement / langage spécifique pour chaque plateforme constitue le nœud du problème, les différentes solutions multiplateformes consistent généralement en des outils sur un langage ou une combinaison de langages génériques et populaires, qui vont servir d’interface avec les fonctionnalités de l’appareil mobile.

Le concept générale (1/3)

Explication du terme “hybride”

- Le développeur n’ayant plus à se soucier de connaître les outils spécifiques à chaque plateforme en développement plusieurs applications, il lui suffit de maîtriser ceux offerts par la solution choisie qui lui compilera une application native spécifique à chaque plateforme à partir de son unique projet de départ.

- PhoneGap (Apache Cordova) fût notemment l’un des pionniers du développement mobile multiplateforme, proposant le populaire trio de langage web HTML5/CSS3/ JavaScript comme source pour ses applications

Le concept général (3/3)

Explication du terme “hybride”

- L’approche utilisée par PhoneGap s’apparentant à une hybridation entre application web et native, le terme "hybride" s’est par la suite démocratisé pour désigner globalement toute application développée avec une solution multiplateforme, il est donc important de noter qu’une "application hybride" n’est pas forcément développée avec une solution exploitant "l’approche hybride".

Les approches

- Si les applications hybrides sont ultimement toutes des applications natives du point de vue du système d'exploitation et de l'utilisateur, il existe effectivement différentes variantes dans la manière dont est créée l'application finale selon la solution utilisée

L’approche hybride (1/3)

- L’approche hybride, utilisée notamment par le moteur Apache Cordova (et donc PhoneGap), vise à tirer parti des objets de type WebView (tels UIWebView sur iOS et android.webkit.WebView sur Android) disponibles nativement sur le système d’exploitation.

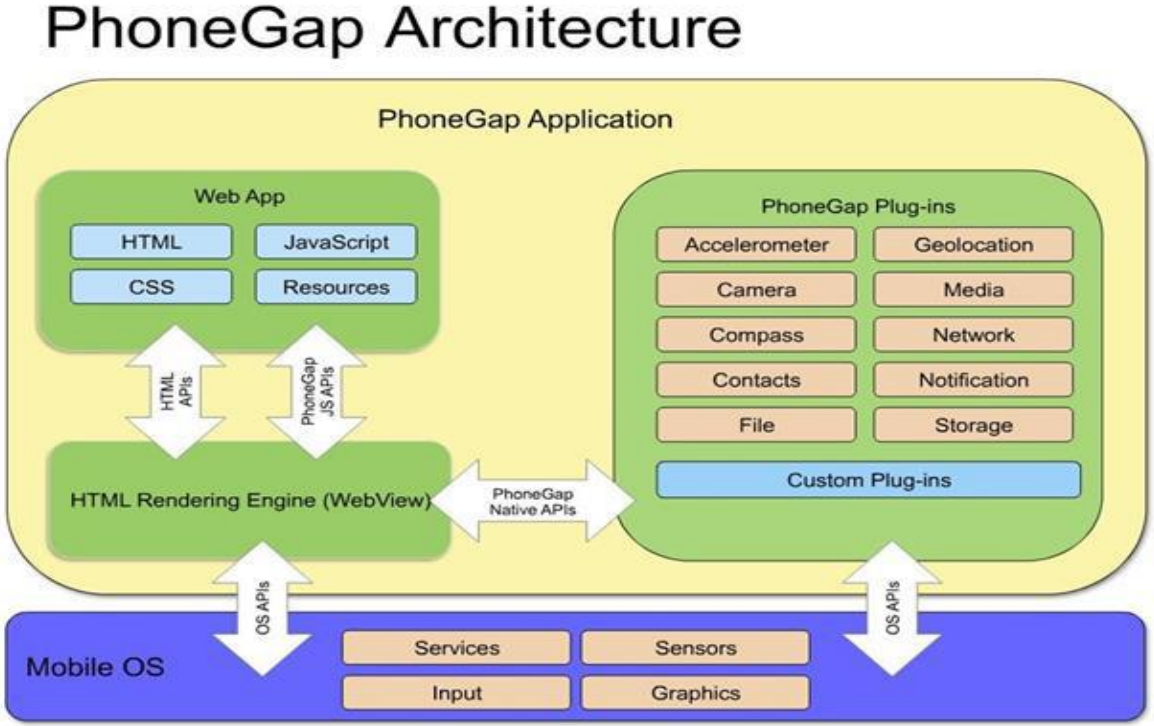
- Les solutions exploitant cette approche compilent de manière autonome une application native spécifique à la plateforme ciblée, enveloppant l’application de départ et dont l’intégralité de l’interface utilisateur consiste en une WebView.

L’approche hybride (2/3)

- Celle-ci va ensuite interpréter, à la manière d’un navigateur, le code et langage web du projet qui aura été développé tout à fait comme s’il s’agissait d’une application web standard. L’accès aux fonctionnalités de l’appareil se fait grâce à des interfaces fournies dans le langage de développement, le développeur n’a donc pas ou très peu à se préoccuper de la plateforme cible, les librairies de la solution gérant les interactions avec l’environnement natif.

L'approche hybride (3/3)

Diagramme d'architecture d'une application PhoneGap



L’approche compilation direct

- Contrairement à l’approche hybride, enveloppant un langage web qui sera interprété à l’exécution de l’application finale, l’approche compilation direct tire parti d’un compilateur spécialisé propre à la solution.

- Celui-ci est capable de créer directement le code assembleur natif spécifique à l’architecture ciblée à partir d’un langage de départ unique (et donc pas forcément web).

- L’avantage par rapport à l’approche hybride est une performance plus élevée, l’architecture étant plus proche d’une réelle application native. Il faut cependant tenir compte du fait qu’un tel système va typiquement nécessiter un environnement d’exécution (runtime) pour l’application finale, celui-ci exposant les API permettant l’accès aux fonctionnalités système.

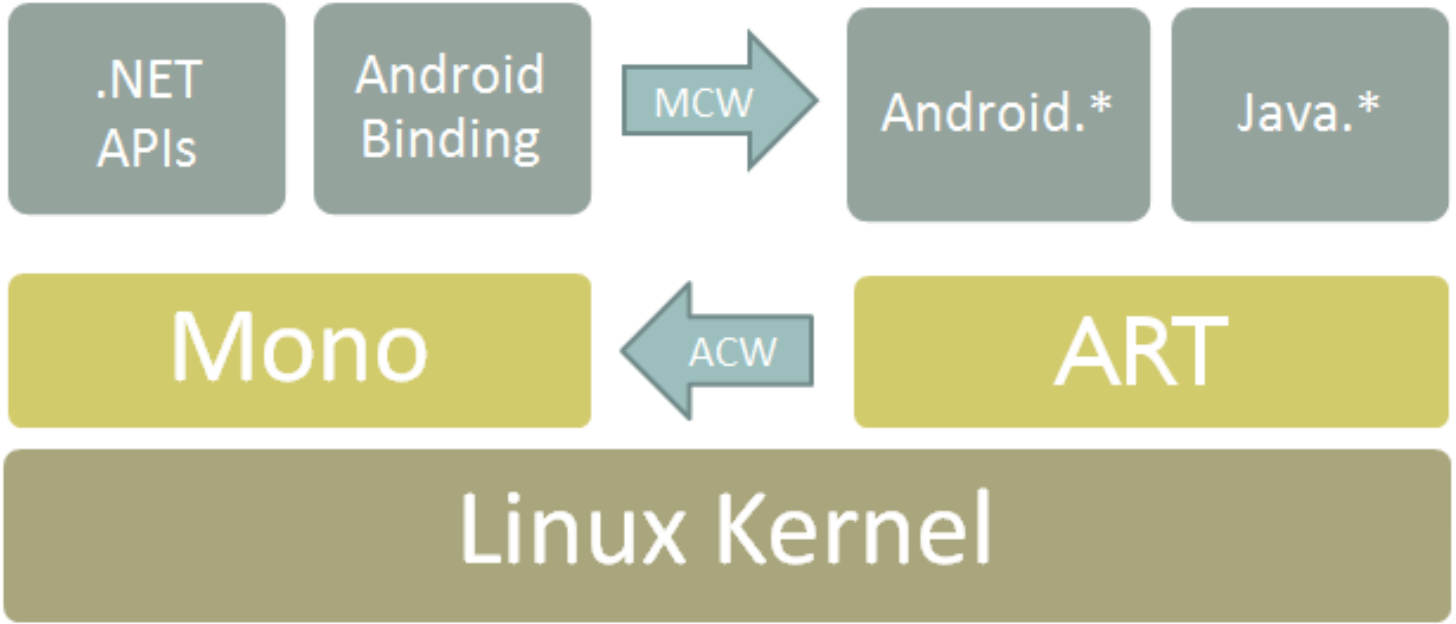
L’approche compilation direct

- Cette approche est typiquement utilisée par Xamarin, les applications fonctionnant à l’intérieur du runtime Mono.

- Le schéma suivant montre l’architecture d’une telle application sur une plateforme Android : comme nous pouvons le voir, l’environnement Mono accède au Kernel Linux et l’expose aux applications Xamarin (C#) de la même manière que l’Android Runtime (ART) permet aux applications Androids standards (Java) de fonctionner

L’approche compilation direct

Diagramme d’architecture, runtime Android - Xamarin



Quelques variantes

- Le système développé par Apache Cordova étant en licence libre, de nombreuses solutions l'exploitant existent et rencontrent généralement beaucoup de succès auprès des développeurs. Apache Cordova en tant que tel n'étant pas un framework de développement front-end et n'étant pas lié à un IDE particulier, une grande partie des solutions l'exploitant vise à le complémenter sur ces points.

- Ionic est l'une des plus populaires, proposant un framework de développement d'applications mobiles HTML, JavaScript et CSS complet tout en étant dérivé du très populaire framework Model-View-Controller, AngularJS.

Les acteurs principaux

Apache Cordova / Adobe PhoneGap

- PhoneGap est un framework de développement mobile multi plateforme initialement développé par Nitobi, entreprise par la suite rachetée par Adobe Systems en 2011.

- Parallèlement à ce rachat, le code du projet fût cédé à la Fondation Apache et la solution renommée Apache Cordova, Adobe Systems l'utilisant par la suite pour proposer ses propres services Adobe PhoneGap puis, plus récemment, Adobe PhoneGap Build.

- Apache Cordova exploite l'approche hybride pour permettre la création d'applications natives à partir de projets web HTML/CSS/ JavaScript. Une très large majorité des plateformes sont supportées tel qu'Android, iOS, Windows Phone, BlackBerry, LG webOS, Firefox OS, Tizen, Ubuntu Touch, Bada et Symbian OS.

Les acteurs principaux

Appcelerator Titanium Mobile

- Appcelerator est une entreprise américaine spécialisée dans les technologies mobiles et fondée en 2006.

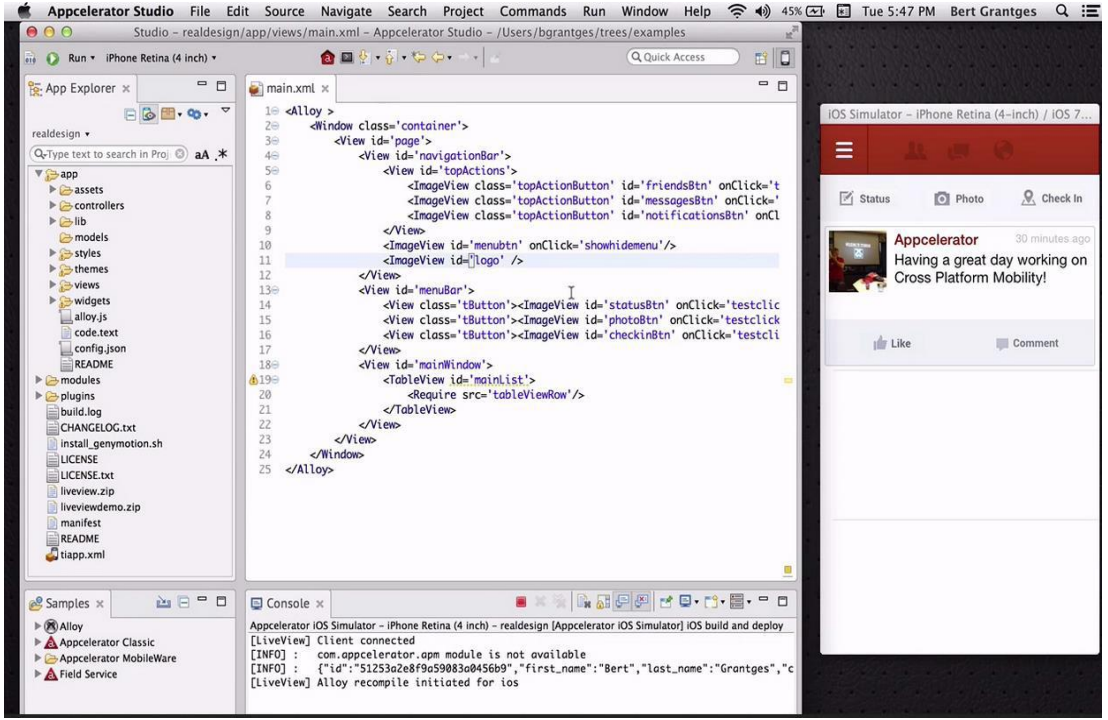
- Elle lança la première version de son framework open source, Appcelerator Titanium, fin 2008, ciblant au départ le développement d'applications multiplateforme pour ordinateurs de bureau. L'entreprise commença ensuite à concentrer son activité sur les applications mobiles en ajoutant le support pour développement iPhone et Android l'année suivante puis en lançant son propre environnement de développement Appcelerator Studio en 2010.

Les acteurs principaux

Appcelerator Titanium Mobile

Appcelerator Studio
offre la possibilité de développer
des applications mobiles
multiplateformes à partir de
JavaScript, le code source de
l'application étant interprété
par un moteur JavaScript à
l'exécution.

.



Les acteurs principaux

Xamarin

- Xamarin est une entreprise américaine fondée en 2011 par l'équipe d'ingénieurs fondatrice du projet open-source Mono

- Actuellement, Xamarin permet de développer des applications mobiles multi-plateforme à partir du langage C# et cela de différentes manières : Son set d'outils Xamarin.Forms permet la plus grande réutilisabilité de code pour des applications exigeant peu de fonctionnalités spécifiques aux plateformes cibles tandis que Xamarin.iOS et Xamarin.Android promettent des interactions plus spécialisées et un accès plus direct aux API spécifiques des différentes plateformes.

IONIC (1/2)

Qu'est-ce que ionic framework ?

- *Ionic* est un framework de développement qui permet de créer des applications hybrides en HTML5, CSS, Javascript

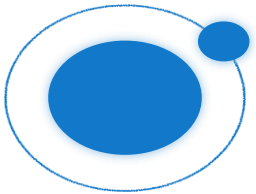
- Il est basé sur des frameworks/technologies qui ont fait leurs preuves telles que AngularJS et Apache Cordova

Angular

+

Cordova

=



IONIC

IONIC (2/2)

Comment Ionic a utilise AngularJS etapache cordova ?

- Ionic est développé avec AngularJS, il est utilisé pour l'implémentation de la partie applicative web (backend). Apache Cordova permet, quant à lui, la construction des applications natives. C'est un pont de développement qui permet d'encapsuler du code client Web (ici Ionic/AngularJS) dans une application native telle que Android, IOS ou encore Windows Phone. Il permet par ailleurs d'avoir certains accès aux fonctionnalités natives des appareils tel que la caméra ou l'accéléromètre

IONIC INSTALLATION (1/3)

Installation de nodeJS

- *Ionic* s'appui sur la plateforme *Node JS*. Il est donc indispensable de l'installer si vous ne l'avez pas encore
- Vous trouverez, sur le lien suivant, le site officiel ou vous pouvez installer *Node JS* :

  https://nodejs.org/en/download/

- *Remarque :* Sous Windows la commande *npm* ne fonctionnera pas sans les variables d'environnements de système JAVA_HOME, ANDROID_HOME et NPM. Sous Unix, les variables d'environnement JAVA_HOME et ANDROID_HOME sont nécessaires.

IONIC INSTALLATION (1/3)

Installation de apache cordova

- Exécuter la suivante commande pour installer *Apache Cordova*. Si vous l'avez déjà

```
$ npm install –g cordova
```

IONIC INSTALLATION (3/3)

Installation de Ionic

- Installez maintenant *Ionic* avec cette ligne de commande

```
$ npm install -g  ionic
```

IONIC et SDK

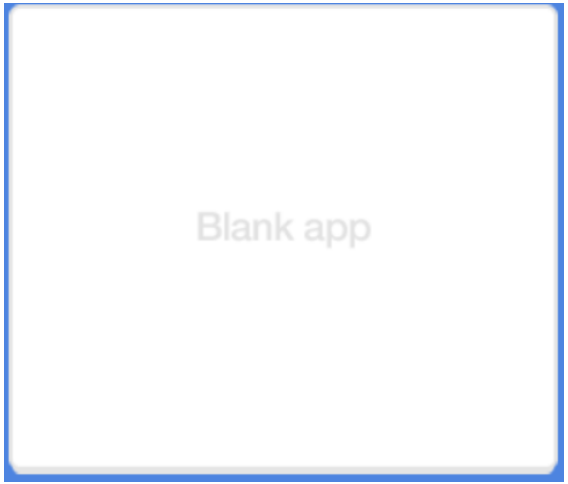
Android SDK

- Pour développer une application pour appareils Android il est indispensable de disposer des outils Android tel que le SDK Android. Si l'on souhaite cibler l'application pour d'autres systèmes d'exploitations, il est là aussi indispensable de disposer du SDK de la plateforme visée. Ceux-ci serviront à la compilation et à la construction de l'application.

IONIC CREATION DE PROJET

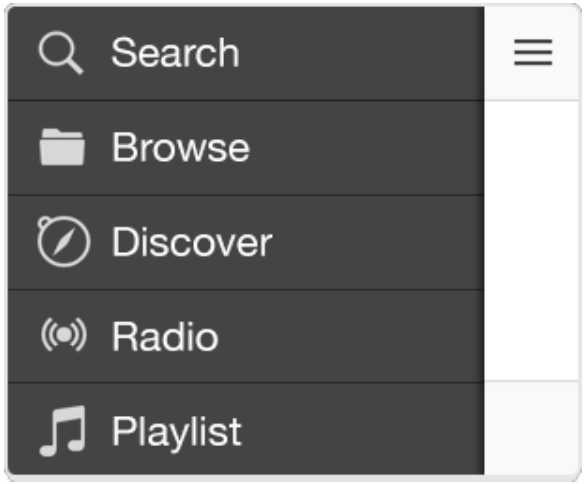
Créer un projet

- Une fois les outils installés, et qu'on dispose d'un environnement de travail opérationnel, il est maintenant temps de créer notre projet que l'on nommera 'uvsApp'.



$ ionic start uvsApp blank

$ ionic start uvsApp tabs



$ ionic start uvsApp sidemenu

Plateformes et déploiement

- **Android**
  - Pour configurer votre application pour les appareils Android. Il suffit d'exécuter la suivante commande à la racine de votre projet

```
$ ionic platform add android
```

  - Ensuite, pour compiler puis lancer et déployer votre application sur un appareil connecté ou depuis un émulateur, il suffit d'exécuter la commande suivante :

```
$ ionic run android
```

Plateformes et déploiement

- IOS
  - Pour configurer votre application pour les appareils IOS. Il suffit d'exécuter la suivante commande à la racine de votre projet :

```
$ ionic platform add ios
```

  - Ensuite, pour compiler puis lancer et déployer votre application sur un appareil connecté ou depuis un émulateur, il suffit d'exécuter la commande suivante :

```
$ ionic run ios
```


Questions ?