TECNOLOGIE WEB MOBILE

Monsieur RANDRIANOMENJANAHARY
Lala Ferdinand
Docteur en Informatique

Définition

La technologie Web Mobile désigne l'ensemble des outils, frameworks et techniques utilisés pour développer des applications web optimisées pour les appareils mobiles (smartphones, tablettes). Ces technologies permettent de créer des sites et applications accessibles via un navigateur web mobile sans nécessiter d'installation depuis un store.

Un store d’un appareil mobile est une plateforme en ligne permettant de télécharger, acheter et mettre à jour des applications mobiles. Il sert d’intermédiaire entre les développeurs et les utilisateurs et garantit la sécurité et la compatibilité des applications avec les appareils mobiles.

I-NATIF

1) MOBILE

-Langage de programmation lié au plateforme

.Android (Java)

.IOS (Objectif-C)

.Windows (C++, C#, VB)

2)WEB

-HTML, CSS, Javascript

II-CROSS-PLATEFORME (HYBRIDE)

-Framework :

.Ionic (Android)

.Flutter (Créer par Google, pour développer des applications pour Android, IOS, Linux, Mac, Windows, Google, Web)

.Corona SDK (Pour créer des applications et des jeux, C++/OpenGL)

.Framework7 (Android)

.React Native (Basé sur React et Javascript)

.Cordova (Android, IOS)

. Appcelerator Titanium (Android et IOS)

. PhoneGap (Android, IOS, Windows phone)

Partie I- Développement Mobile Multiplateformes

Chapitre 1. Développement mobile multiplateformes

1.1 Quelles sont les usages courants ?

- Jeux vidéos
- Applications Grand publique
- Applications Professionnelles

1.2.Forme du code exécutable sur un appareil mobile

- Natif

  -Android (Java), IOS(Objectif-C), Windows(C++)
- Web

  -HTML, CSS, JavaScript

1.3.Avantages du code natif

- Intégration fine
- Interface et ergonomie
- Haute performance
- Outillage
- Markets

1.4.Inconvénients du code natif

- Plateforme spécifique
- Temps et argent
- Problème de ressources

1.5.Avantages des applications web sur appareil mobile

- Plusieurs plateformes
- Bonne interaction
- Performance
- IHM professionnelle
- Gain de temps / argent

1.6.Inconvénients des applications web sur appareil mobile

- Pas de hautes performances
- Pas d’intégration

1.7.Comparaison entre natif et WebMobile

Natif

- IHM
- Hautes performances
- Outillages
- Markets

WebMobile

- Cross-platforms
- Gain de temps / argents
- Ressources

Chapitre 2. WebMobile Hybride

2.1.Qu’est-ce que c’est ?

- Consiste généralement en des outils basés sur un langage ou une combinaison de langages génériques et populaires, qui vont servir d’interface avec les fonctionnalités de l’appareil mobile.
- Le développeur n’ayant plus à se soucier de connaître les outils spécifiques à chaque plateforme en développement plusieurs applications,
- Il 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.
- Exemple : PhoneGap (Apache Cordova) est l’un des pionniers du développement mobile multiplateforme, proposant le populaire trio de langage web HTML5/CSS3/ JavaScript comme source pour ses applications.

2.2 Avantages

Permet de :

- Embarquer son application Web Mobile au sein d'une application native
- Bénéficier d'un pont entre le système et notre application Web Mobile
- Accéder aux markets

2.3 Modes de production (génération) de codes exécutables

- Ecrire en natif et générer en natif :

  - API Java Android à Android

  - Objectif-C à IOS

  -C#, VB à WindowsPhone

- Ecrire en HTML, CSS, JAVASCRIPT et générer une application web :

  -Applications Web sur navigateur

- Ecrire en HTML, CSS, JavaScript + Code natif et générer le webHybrid :

  -Exemple Cordova

- Ecrire dans un seul et unique langage et générer en natif :

  -Exemple Xamarin

Partie II- Cordova

3.3 Cordova: La pratique

à l'aide de l'interface de ligne de commande (CLI) cordova, Cordova permet de créer une application Web Mobile et la déployer sur diverses platesformes mobiles natives.

3.4 Installation de CLI Cordova

- L'outil de ligne de commande Cordova est distribué sous la forme d'un package npm.
- Pour installer l'outil de ligne de commande Cordova, procédez comme suit :**1. Téléchargez et installez**.
- Lors de l'installation, vous devriez pouvoir invoquer node et npm sur votre ligne de commande.
- **2.(Facultatif) Téléchargez et installez le client git** si vous n'en avez pas déjà un. Après l'installation, vous devriez pouvoir invoquer git sur votre ligne de commande. La CLI l'utilise pour télécharger des actifs lorsqu'ils sont référencés à l'aide d'une URL vers un référentiel git.
- **3. Installez le module cordova** à l'aide de l'utilitaire npm de . Le module cordova sera automatiquement téléchargé par l'utilitaire npm.

- Sous OS X et Linux, $**sudo npm install -g cordova**

- Sous OS X et Linux, il peut être nécessaire de préfixer la commande npm avec sudo pour installer cet utilitaire de développement dans des répertoires autrement restreints tels que /usr/local/share.

sous Windows :

C:\>**npm install -g cordova**

- L'indicateur -g ci-dessus indique à npm d'installer cordova globalement.
- Sinon, il sera installé dans le sous-répertoire node_modules du répertoire de travail courant.
- Après l'installation, vous devriez pouvoir exécuter cordova sur la ligne de commande sans arguments et il devrait imprimer le texte d'aide.

3.5 Créer l'application

Accédez au répertoire dans lequel nous gérons notre code source et créez un projet cordova :

$ cordova create <nom_de_votre_application> <id_de_votre_application> <nom_de_votre_projet>

<nom_de_votre_application> : Remplacez ceci par le nom que vous souhaitez donner à votre application.

<id_de_votre_application> : Ceci est l'identifiant unique de votre application, généralement sous la forme de "com.exemple.nomapp".

<nom_de_votre_projet> : Le nom du répertoire qui sera créé pour contenir votre projet Cordova.

Exemple Créer l'application MonApp

cordova create monapp fr.exemple.monapp MonApp

cd monapp

Cela crée une structure comme :

```
monapp/
├── config.xml
├── hooks/
├── platforms/
├── plugins/
├── www/            ← Contient les fichiers HTML/CSS/JS
```

Configuration de base (config.xml)

```
<name>MonApp</name>
<description>Application Cordova mobile</description>
<author>Toi</author>
<content src="index.html" />
```

Vous pouvez aussi ajouter des autorisations :

```
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" /
    >
```

Cette ligne déclare une autorisation Android que ton application demande à l'utilisateur lors de l'installation ou au moment de l'utilisation (selon la version d'Android).

Le répertoire hooks/ d'une application Cordova contient des scripts personnalisés que tu peux exécuter automatiquement à différents moments du cycle de vie du projet (création, build, ajout de plateforme, etc.).

À quoi sert le dossier hooks/ ?

Il sert à automatiser certaines tâches en ajoutant des scripts qui s'exécutent à des étapes précises, comme :

-Avant ou après l'installation d’un plugin

-Avant ou après un build (cordova build)

-Avant ou après l’ajout d’une plateforme (cordova platform add)

-Avant ou après la préparation (cordova prepare)

Structure typique

```
hooks/
├── before_build/
├── after_build/
├── before_plugin_add/
├── after_plugin_add/
├── before_prepare/
├── after_prepare/
└── ...
```

Chacun de ces sous-dossiers contient des scripts (JavaScript, Bash, etc.) qui seront exécutés au bon moment.

3.6 Ajouter des plateformes

- Toutes les commandes suivantes doivent être exécutées dans le répertoire du projet ou dans n'importe quel sous-répertoire :

$ cd hello

Ajoutez les plates-formes que vous souhaitez cibler pour votre application. Nous ajouterons les plateformes "ios" et "android" et veillerons à ce qu'elles soient enregistrées dans :

$ cordova add ios –save

$ cordova add android –save

Pour vérifier votre ensemble actuel de plates-formes :

$ cordova ls

- L'exécution de commandes pour ajouter ou supprimer des plateformes affecte le contenu du répertoire des plateformes du projet, où chaque plateforme spécifiée apparaît comme un sous-répertoire.
- Remarque : Lorsque vous utilisez l'interface de ligne de commande pour créer votre application, vous ne devez modifier aucun fichier dans le répertoire /platforms/. Les fichiers de ce répertoire sont systématiquement écrasés lors de la préparation des applications pour la construction ou lorsque les plugins sont réinstallés.

3.7 Installer les pré-requis pour la construction

Pour créer et exécuter des applications, vous devez installer des SDK (Software Development Kit) pour chaque plateforme que vous souhaitez cibler.

➢

Pour installer le SDK pour la plateforme Android, suivez ces étapes :

- Téléchargez et installez Android Studio : Allez sur le site officiel d'Android Studio (https://developer.android.com/studio) et téléchargez la version correspondant à votre système d'exploitation. Suivez les instructions d'installation une fois le téléchargement terminé.

- Lancez Android Studio : Une fois installé, lancez Android Studio. La première fois que vous lancez Android Studio, il peut télécharger et installer certains composants nécessaires, comme le SDK Android.

- Ouvrez le SDK Manager : Une fois Android Studio lancé, cliquez sur "Configure" dans la barre d'outils, puis sélectionnez "SDK Manager".

-Sélectionnez les packages à installer : Dans le SDK Manager, vous verrez une liste de packages à installer. Assurez-vous que la case à cocher à côté de "Android SDK Platform" pour la version Android que vous souhaitez cibler est cochée. Vous pouvez également installer d'autres packages comme les outils de build, les images système, selon vos besoins.

-Cliquez sur "Apply" ou "OK" : Une fois que vous avez sélectionné les packages à installer, cliquez sur le bouton "Apply" ou "OK" pour démarrer le téléchargement et l'installation des packages sélectionnés.

-Une fois que les packages sont installés, vous aurez configuré le SDK Android sur votre système. Vous pourrez maintenant utiliser Cordova pour ajouter la plateforme Android à votre projet et développer des applications pour cette plateforme.

Pour vérifier si vous remplissez les conditions requises pour créer la plateforme :

$ cordova requirements

Résultats de la vérification des exigences pour Android :

Java JDK: installed .

Android SDK: installed

Android target: installed android-19,android-21,android-22,android-23 ,Google Inc.:Google APIs:19,Google Inc.:Google APIs (x86 System Image

):19,Google Inc.:Google APIs:23

Gradle: installed

Résultats de la vérification des exigences pour ios :

Apple OS X: not installed

Cordova tooling for iOS requires Apple OS X

Error: Some of requirements check failed

Gradle Android

C’est un plugin spécifique pour Gradle, un système de build open-source largement utilisé dans le développement logiciel, qui est principalement utilisé pour la construction de projets Android. Gradle est conçu pour automatiser le processus de construction, de test et de déploiement des projets logiciels, et il est hautement configurable et extensible.

Le plugin Gradle Android fournit des fonctionnalités spécifiques pour le développement d'applications Android. Voici quelques-unes de ses principales fonctionnalités :

1-Gestion des dépendances : Gradle Android permet de spécifier les dépendances de votre projet Android, telles que les bibliothèques tierces ou les modules internes.

2-Configuration du projet : Vous pouvez configurer divers aspects de votre projet Android, tels que les versions minimales et cibles d'Android, les configurations de build, les options de compilation, etc.

3-Construction de différentes variantes : Gradle Android facilite la création de différentes variantes de votre application, par exemple, des versions de débogage et de production, des versions pour différentes architectures de processeurs, etc.

4-Intégration avec les outils Android : Le plugin Gradle Android s'intègre étroitement avec les outils de développement Android, tels que l'outil de compilation de ressources, le compilateur Java, le générateur de ressources R, etc.

5-Gestion des ressources : Vous pouvez gérer efficacement les ressources de

3.8 Construire l'application

- Par défaut, cordova create script génère une application Web squelettique dont la page de démarrage est le fichier du projet.
- .
- Exécutez la commande suivante pour créer le projet pour toutes les plateformes :

$ cordova build

- Nous pouvons éventuellement limiter la portée de chaque build à des plateformes spécifiques - "ios" dans ce cas :

$ cordova build ios

3.9 Testez l'application

- Exécutez une commande telle que la suivante pour reconstruire l'application et l'afficher dans l'émulateur d'une plateforme spécifique :

$ cordova emulate android

- Le suivi de la commande cordova emulate actualise l'image de l'émulateur pour afficher la dernière application, qui est maintenant disponible pour le lancement depuis l'écran d'accueil :

Alternativement, vous pouvez tester directement l'application :
$ cordova run android

3.10 Ajouter des plugins

- Vous pouvez modifier l'application générée par défaut pour tirer parti des technologies Web standard, mais pour que l'application accède aux fonctionnalités au niveau de l'appareil, vous devez ajouter des plugins.
- Un plugin expose une API Javascript pour la fonctionnalité SDK native.
- Les plugins sont généralement hébergés sur npm et vous pouvez les rechercher sur la page de recherche de plugins.
- Certaines API clés sont fournies par le projet open source Apache Cordova et sont appelées API Core Plugin. Vous pouvez également utiliser la CLI pour lancer la page de recherche :

```
$ cordova plugin search camera
```

- Pour ajouter le plugin caméra, nous allons spécifier le nom du package npm pour le plugin caméra :

```
$ cordova plugin add cordova-plugin-camera
```

- REMARQUE : L'interface de ligne de commande ajoute le code de plugin approprié pour chaque plateforme.

- Utilisez plugin ls (ou plugin list, ou plugin par lui-même) pour voir les plugins actuellement installés. Chacun affiche par son identifiant :

```
$ cordova plugin ls
cordova-plugin-camera 2.1.0 "Camera" cordova-plugin-whitelist 1.2.1 "Whitelist"
```

4.12 Mise à jour de Cordova et de votre projet

- Après avoir installé l'utilitaire cordova, vous pouvez toujours le mettre à jour vers la dernière version en exécutant la commande suivante :

$ sudo npm update -g cordova

- Utilisez cette syntaxe pour installer une version spécifique :

$ sudo npm install -g cordova@3.1.0-0.2.0

- Exécutez cordova -v pour voir quelle version est en cours d'exécution. Pour trouver la dernière version cordova publiée, vous pouvez exécuter :

$ npm info cordova version

- Pour mettre à jour la plate-forme que vous ciblez :

$ cordova platform update android --save

$ cordova platform update ios –
save

4.13 Configuration requise et support

- Cordova pour Android nécessite le SDK Android qui peut être installé sur le système d'exploitation OS X, Linux ou Windows.

- Cordova supporte Android 4.0.x (en commençant par le niveau de l'API Android 14) et plus élevé.

4.14 Installer les outils de Cordova Shell

- Si vous souhaitez utiliser les outils de coquille Android de Cordova conjointement avec le SDK, Télécharger Cordova Sinon ignorer cette section si vous envisagez d'utiliser l'outil CLI de multi-plateforme décrit dans l'Interface de ligne de commande.

- Le téléchargement de Cordova contient des archives distincts pour chaque plateforme. N'oubliez pas d'élargir l'archive appropriée, android dans ce cas, dans un répertoire vide. Les utilitaires pertinents sont disponibles dans le niveau supérieur bin répertoire. (Consultez le fichier **README** si nécessaire pour des directions plus détaillées).

- Ces outils de coquille permettent de créer, générer et exécuter des applications Android. Pour plus d'informations sur l'interface de ligne de commande supplémentaire qui active les fonctionnalités de plugin sur toutes les plateformes, voir Plugman pour l'aide à gérer les Plugins. Voir Plugins Application pour plus d'informations sur la façon de développer des plugins.

5.2 Installer les paquets SDK

- Ouvrez le gestionnaire de SDK Android et installer :

1. 5.1.1 Android (API 22) platform SDK
2. Selectionner la Version d'Android SDK Build-tools

5.3 Configurer un émulateur

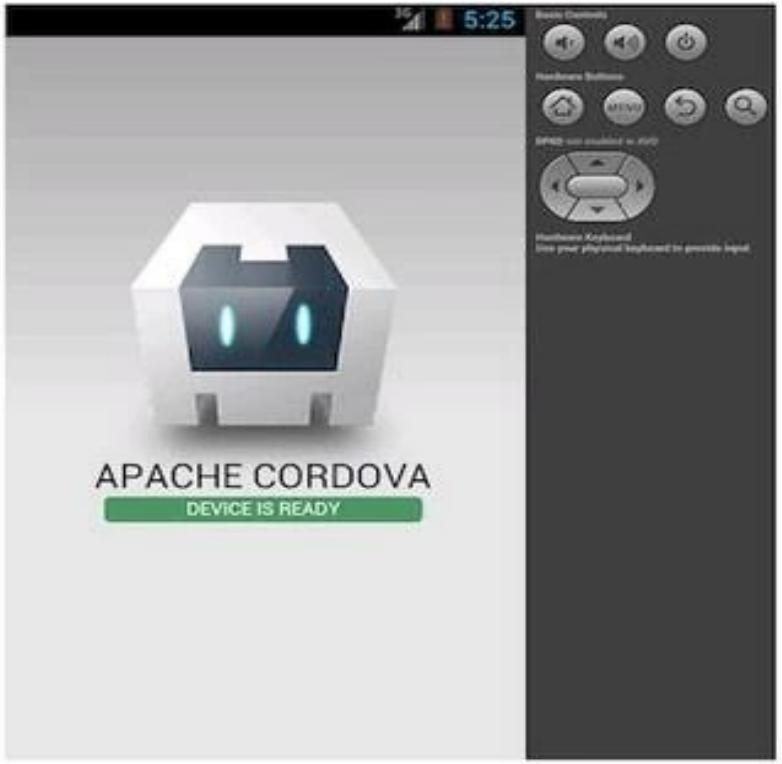
- Android sdk ne fournit pas de n'importe quelle instance d'émulateur par défaut. Vous pouvez créer un nouveau en exécutant android sur la ligne de commande. Cliquez sur **Outils puis gérer AVDs** (périphériques virtuels Android), puis choisissez n'importe quel élément du **Dispositif de définitions** dans la boîte de dialogue :

- Appuyez sur **Créer AVD**, éventuellement modifier le nom, puis appuyez sur **OK** pour accepter les modifications :
- L'AVD apparaît alors dans la liste **Des périphériques virtuels Android** :

Pour ouvrir l'émulateur comme une demande distincte, l'AVD et cliquez sur **Démarrer**. Il lance autant qu'il le ferait sur le dispositif, avec des contrôles supplémentaires disponibles pour les boutons matériels modernes fournissent des extensions pour exécuter des Machines virtuelles plus efficacement.:

- Avant d'utiliser ce type d'accélération, vous devez déterminer quel est le CPU de votre système actuel de développement, on supporte les technologies de virtualisation suivants :


5:25
APPS
WIDGETS
Browser
Calculator
Calendar
Camera
Clock
Custom Locale
Dev Settings
Dev Tools
Downloads
Gallery
Hello World
Messaging
Music
People
Phone
Search
Settings
Speech Recorder


5:25
APACHE CORDOVA
DEVICE IS READY

- **Technologie de virtualisation Intel** (VT-x, vmx) ?-x pris en charge la liste des processeurs

- **AMD Virtualization** (AMD-V, SVM), pris en charge uniquement pour Linux (depuis mai 2006, tous les processeurs AMD incluent AMD-V, sauf Sempron).

- Une autre façon de savoir si votre processeur supporte la technologie de VT-x, c'est en exécutant l' Utilitaire Intel Processor Identification Utility, pour Windows, vous pouvez le télécharger depuis le Centre de téléchargementde Intel, ou vous pouvez utiliser l'utilitaire booteable, qui est Indépendant de l'OS.

- Après le téléchargement, exécuter le programme d'installation d'Intel, qui est disponible dans votre Android SDK àOptions/intel/Hardware_Accelerated_Execution_Manager.

5.5 Générez le projet

- Si vous utilisez l'interface CLI dans le développement, le répertoire de niveau supérieur www du répertoire du projet contient les fichiers sources.

```
$ cordova build                 # build all platforms that were added
$ cordova build android         # build debug for only Android
$ cordova build android --debug   # build debug for only Android
$ cordova build android --release # build release for only Android
```

- Si vous utilisez les outils de coquille spécifiques à Android en développement, il y a une approche différente. Une fois que vous générez le projet, source de l'application par défaut est disponible dans le sous-répertoire assets/www . Les commandes suivantes sont disponibles dans son sous-répertoire de cordova .
- La commande build nettoie les fichiers projet et régénère l'app. Voici la syntaxe pour Mac et Windows. Les deux premiers exemples génèrent des informations de débogage, et le second s'appuie les apps pour diffusion immédiate :

```
 $ /path/to/project/cordova/build --debug
C:\path\to\project\cordova\buid --debug
$ /path/to/project/cordova/build --release
C:\path\to\project\cordova\buid --release
```

5.6 Déployer l'application

- Vous pouvez utiliser l'utilitaire CLI de cordova pour déployer l'application sur l'émulateur ou le dispositif de la ligne de commande :
- **-Pour déployer l'application dans un émulateur Android par défaut**

$ cordova emulate android

-Pour déployer l'application dans un appareil connecté

$ cordova run android --device

Lancer l’app Cordova sur l’émulateur

Option A (Android Studio GUI) :
1-Clique sur Run
2-Sélectionne ton émulateur actif
3-Android Studio build et exécute l’app Cordova

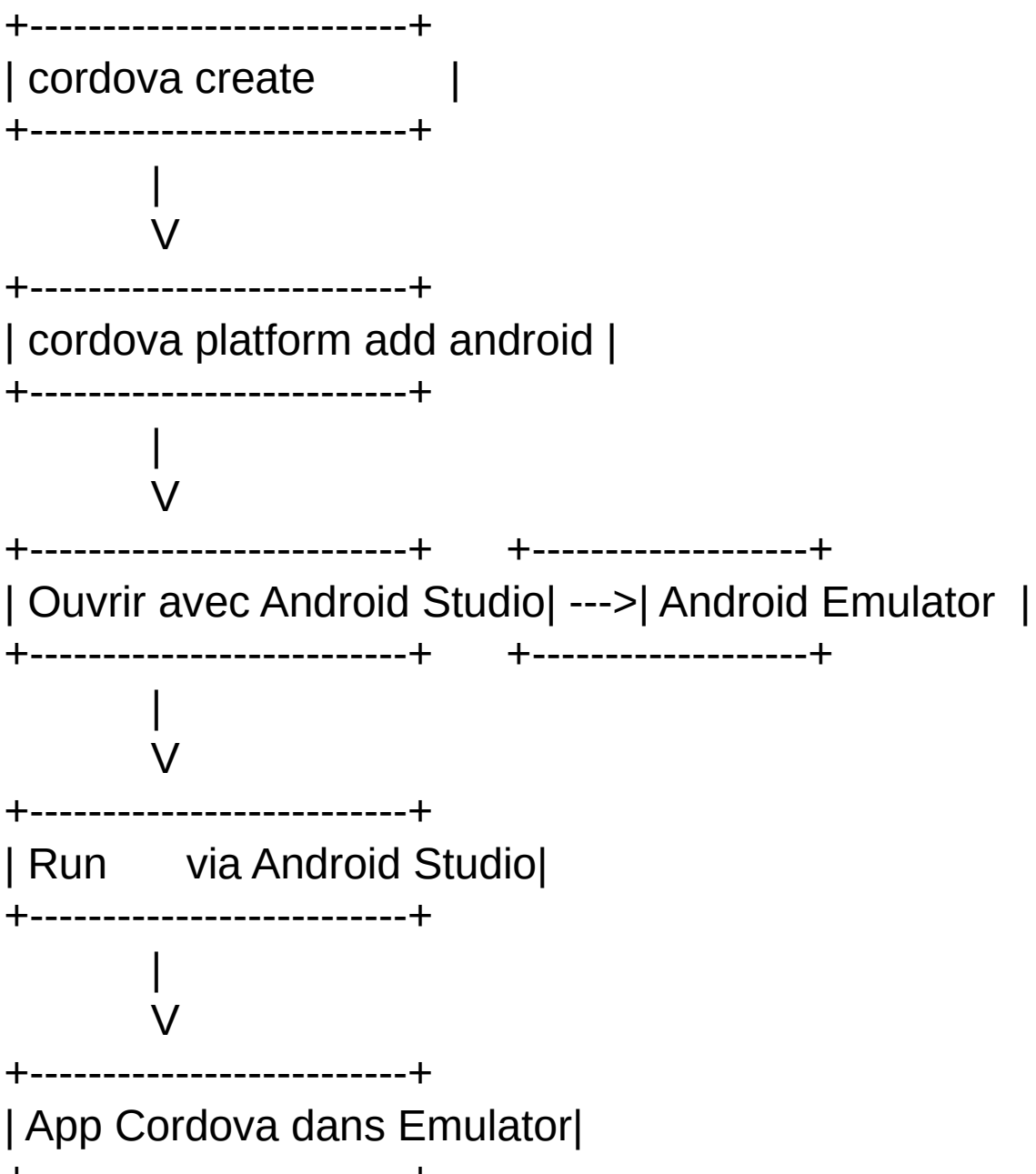
Option B (CLI, si tu préfères) :
cordova emulate android

Cela détectera automatiquement l’émulateur actif et installera l’application dessus.

Astuce : Modifier l’interface Cordova

Pour modifier le contenu affiché dans l’application :
cd cordovaDemo/www
Modifie index.html, index.js, etc. → Ensuite relance l'app.

Schéma complet du processus



5.6 Déployer l'application

-Voir toutes les cibles disponibles

$ cordova run android –list

-Pour exécuter l'application sur un émulateur ou un périphérique spécifique

$ cordova run android –target=target_name

Exemple :

$ cordova run android -target="Nexus4_emulator"

Cela pousse l'application à l'écran d'accueil et il se lance

-Pour voir les aides.

$ cordova run --help

La commande **cordova run --debug** permet de compiler et lancer une application Cordova sur un émulateur ou un appareil physique en mode debug, ce qui est très utile pendant le développement.

Syntaxe complète

cordova run [platform] –debug

Paramètres

run : construit le projet et l’installe/lance sur un appareil ou un émulateur.

platform : la plateforme cible (android, ios, etc.)

--debug : construit le projet en mode développement (debug) avec tous les outils de débogage activés.

Ce que fait --debug concrètement

Fonction	Description
Compilation non optimisée	Garde les fichiers lisibles, sans minification ni obfuscation.
Ajout de WebView debug	Active l’accès aux outils de développement (chrome://inspect pour Android).
Génère un APK .apk ou .ipa	Génère un build debug dans le dossier /platforms/<platform>/app/build/outputs/apk/debug/.
Installation automatique	Déploie l’application sur un appareil/emulateur connecté.
Débogage en temps réel	Permet le live reload, l’inspection DOM, le JS debugging avec Chrome DevTools.

Exemple d’utilisation

Pour Android :
cordova run android --debug
Cela :

-Compile ton application Cordova pour Android en mode debug.

-Lance l’émulateur ou utilise un appareil connecté.

-Installe l’app en version de développement.

À noter
Par défaut, Cordova utilise --debug si aucun --release n’est précisé.

Tu peux ajouter --device pour cibler un appareil physique :
cordova run android --debug --device

Commande cordova run --nobuild expliquée dans Cordova

La commande cordova run --nobuild est utilisée pour exécuter une application Cordova sur un appareil ou un émulateur, sans reconstruire le projet. Elle suppose que l'app a déjà été construite auparavant.

Syntaxe

cordova run [platform] –nobuild

Paramètres

Paramètre	Description
run	Exécute l'application sur un appareil ou un émulateur.
platform	Plateforme ciblée (android, ios, etc.)
--nobuild	Utilise les fichiers de build déjà générés, sans recompiler.

Utilisation typique

cordova run android --nobuild

Cela :

Ne reconstruit pas les fichiers source (.js, .html, plugins, etc.).

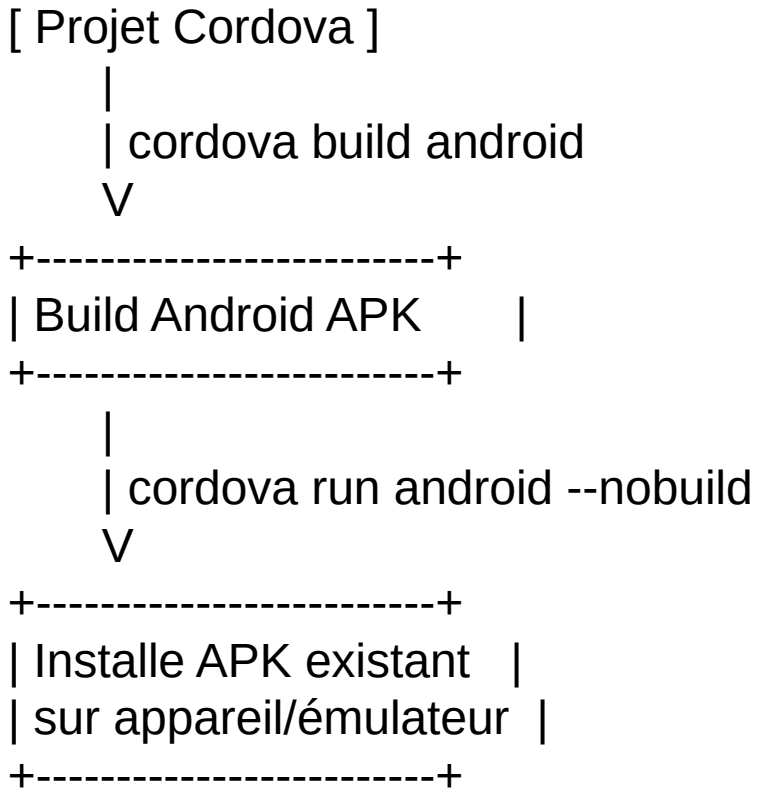
Utilise le dernier APK ou build .apk/.ipa généré dans platforms/android/app/build/outputs/apk/debug/.

Lance directement l’application sur un appareil ou un émulateur.

Quand utiliser --nobuild ?

Cas d’usage	Pourquoi c’est utile
Tu as déjà compilé avec cordova build	Pas besoin de recompiler inutilement.
Tu fais des tests rapides de lancement / comportement	Gagne du temps, surtout si aucun fichier n’a changé.
Tu modifies uniquement des fichiers natifs (Java/Kotlin/Obj-C)	Cordova ne les reconstruit pas de toute façon.
Tu veux tester différents appareils avec le même build	Recycle le build existant.

Schéma du processus avec --nobuild



Exemple combiné

cordova build android
cordova run android –nobuild

→ Le premier compile, le second exécute le même build sans recompilation.

5.7 Autres commandes

- Ce qui suit génère un journal détaillé de l'application en cours d'exécution :

$ /path/to/project/cordova/log

C:\path\to\project\cordova\log

- Le texte suivant nettoie les fichiers de projet :

$ /path/to/project/cordova/clean

C:\path\to\project\cordova\clean

5.8 Ouvrez un nouveau projet dans le SDK

- Une fois que la plateforme android est ajouté à votre projet, vous pouvez l'ouvrir depuis AndroidStudio :

1. Lancez l'application **Android de Studio** .
2. Sélectionnez **Import Project (Eclipse ADT, Gradle, etc.)**.
3. Sélectionnez l'emplacement où la plateforme android est stockée (votre/projet/platforms/android).
4. Pour la question Gradle Sync vous pouvez simplement répondre **Oui**.

- Vous pouvez générer et exécuter l'application directement à partir de Studio Android.