Développement Web mobile

Partie 1 : développement d'applications hybrides avec Apache Cordova

Partie 2 : Ionic

Développement hybride

- Développement Web
- Développement Natif : spécifique à un OS
  - Android (Java, Kotlin, etc.)
  - iOS (Swift)
  - WindowsPhone (RIP 12/2019)

https://www.developpez.com/actu/241967/Microsoft-recommande-aux-utilisateurs-de-son-OS-Windows-10-Mobile-de-passer-a-iOS-ou-Android-la-fin-du-support-de-ce-dernier-approchant-a-grands-pas/

→ Code à développer *n* fois

Développement hybride

- Développement Hybride : mélange de
  - Technologies Web HTML, CSS, JS
  - Technologies natives mobiles pour utiliser certaines fonctionnalités du smartphone

- Développement *cross-platform* :
  - utilisation d'un *framework* pour générer à partir d'un code unique une application pour chaque OS cible
  - ex :
    - Cordova, PhoneGap à partir de code hybride
    - React Native à partir de code React
    - Visual Studio tools pour Xamarin, à partir de code .net
      https://docs.microsoft.com/fr-fr/xamarin/

Développement cross-platform

- Avantages :
  - un seul développement
  - utilisation des compétences des développeurs web
- Inconvénients :
  - rapidité ?
  - design ?
  - ergonomie ?

Cordova

- Historique :
  - Initialement Adobe PhoneGap
  - 2011 : Apache Cordova

    → Open Source
  - PhoneGap existe toujours : développement de plugins payants
- Fonctionnement :
  - Code html non compilé, mais intégré dans une *WebView* native
  - Moteurs de rendu et d'intégration moins efficaces que du code natif

Installation / première app

- Installation : https://cordova.apache.org/#getstarted

```
npm install -g cordova
```

- Créer un projet :

```
cordova create <projectName>
```

- Configurer un OS cible : `cd projectName` puis

```
cordova platform add android | ios | osx | windows | electron | browser
```

Installation / première app

- Tester :

  `cordova run android` → sur un smartphone branché en USB, sur un émulateur si non trouvé

  `cordova emulate android`

- Cela nécessite :
  - SDK android
  - gradle

Installation / première app

- Plusieurs problèmes peuvent subvenir :
  - Sdk android non trouvé
    - → créer un lancer `run.sh`

```
export ANDROID_SDK_ROOT=/usr/local/Android-Studio-Sdks/

cordova run android
```

puis `sh run.sh`

  - Gradle non trouvé
    - → Machine personnelle : installer gradle

```
sudo apt-get install gradle
```

Installation / première app

- Gradle en F39 n'est pas exécutable…
- Nécessité de le copier

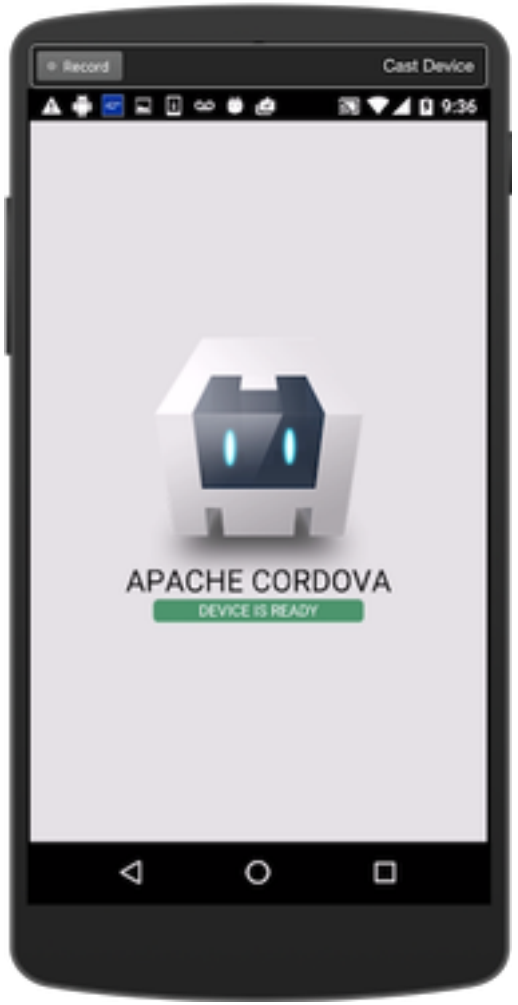
```
cp -r /usr/local/Android-Studio/gradle .

cd gradle/gradle-5.1.1/bin

chmod a+x gradle
```

- Ajouter à `run.sh :`

```
export PATH=./gradle/gradle-5.1.1/bin/:$PATH
```

Installation / première app

- Si tout se passe bien :



Et iOS ?

- Dans ce qui précède, seul Android a été évoqué
- Pour iOS, nécessité d'installer
  - Xcode
  - Le module npm ios-deploy
  - Etc.
  - https://cordova.apache.org/docs/en/latest/guide/platforms/ios/index.html

Structure du projet

- config.xml

  nom, description, auteur de l'appli

  lien vers le fichier d'entrée index.html

  plugins nécessaires

  plateformes cibles

```
./
../
config.xml
hooks/
node_modules/
package.json
package-lock.json
platforms/
plugins/
www/
```

Structure du projet

- www/

  code source html, css, js

- plugins/

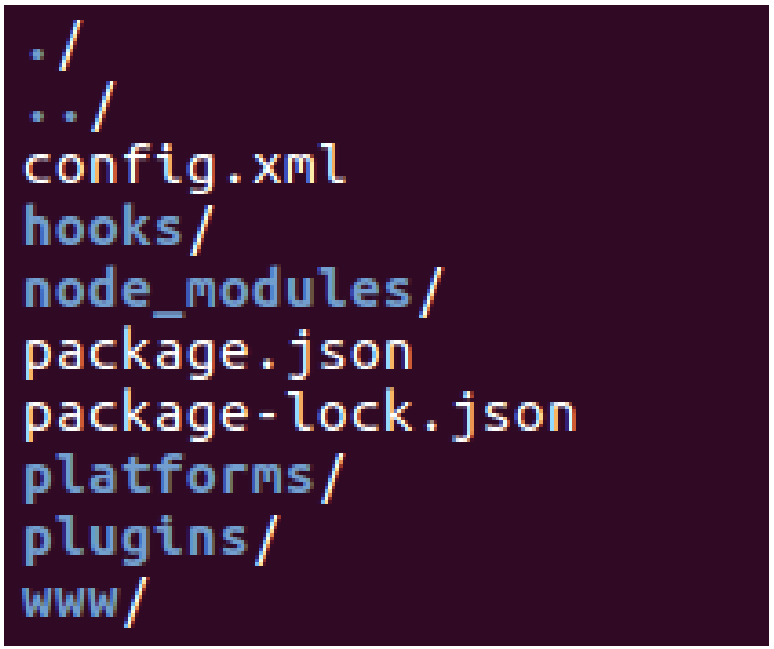
  plugins nécessaires à

  votre application

  installés grâce à

  `cordova plugin add <pluginName>`

  par défaut `cordova-plugin-whitelist`



Structure du projet

- platforms/

  fichiers nécessaires à la

  génération du code cible
- node_modules/

  packages node.js

```
./
../
config.xml
hooks/
node_modules/
package.json
package-lock.json
platforms/
plugins/
www/
```

- hooks/ : scripts optionnels pour personnaliser les scripts cordova

Structure du projet

- package.json

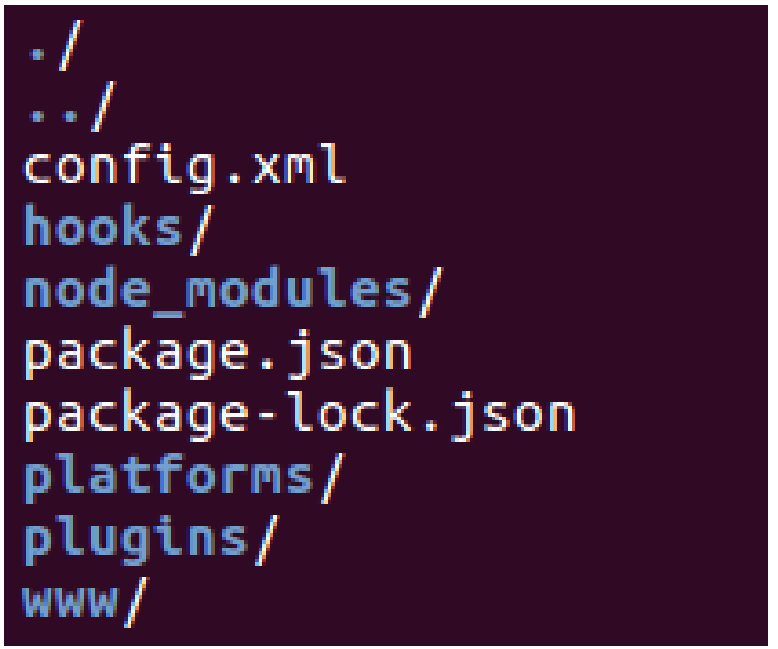
  identification de l'app

  dépendances

  plugins

- package-lock.json

  structure exacte de votre app, avec tous les packages et leur version nécessaires



Structure du projet

- Retour sur www/

  css/

  js/

  img/

  index.html

- Ce qu'on n'y trouve pas :

  PHP !!!!!!!!!!!!!!!!

- Uniquement le côté client !

```
./
../
config.xml
hooks/
node_modules/
package.json
package-lock.json
platforms/
plugins/
www/
```

Cordova.js

- Dans les fichiers html :

  inclusion de cordova.js, qui n'est pas dans js/
- Placé dans platforms/…
- A conserver !

```
<div class="app">
    <h1>Apache Cordova</h1>
    <div id="deviceready" class="blink">
        <p class="event listening">Connecting to Device</p>
        <p class="event received">Device is Ready</p>
    </div>
</div>
<script type="text/javascript" src="cordova.js"></script>
<script type="text/javascript" src="js/index.js"></script>
```

Comment coder ?

- En séparant la couche client de la couche serveur

  → appels au serveur via une API, en ajax (XMLHttpRequest, jquery, axios, etc.)

  → manipulation du DOM en JS pour alimenter le template HTML

- Possible en :
  - JS de base
  - Frameworks (React, Vue, Ionic, etc.)

React + Cordova (1)

- Créer l'app cordova et supprimer les fichiers sources générés

```
cordova create RtoC ; cd RtoC ;

cordova platform add android ; cd www ; rm -rf *
```

- Du côté React :
  - Dans package.json, ajouter

    ```
    "homepage": "./",
    ```

    avant `"dependencies"`
  - Faire le build du projet react

    ```
    npm run build
    ```

React + Cordova (2)

- Revenir du côté cordova, et copier les fichier du dossier `build` de react dans `www`
- Si nécessaire, ajouter l'appel au script cordova.js dans l'index.html
- Faire le run (puis build) cordova

```
cordova run android
```

→ Votre appli React est maintenant une appli mobile

Documentation et test

- https://cordova.apache.org/docs/en/latest/
- Avec notamment la liste des plugins nécessaires / disponibles

- L'app peut être testée sur émulateur ou smartphone
- Mais aussi dans un navigateur avec file://

  ou avec `cordova run browser`

Build

- Une fois l'application terminée, création de l'app pour l'OS cible

```
export ANDROID_SDK_ROOT=/home/laroche5/android-sdks/

cordova build --info –release
```

- En option :
  - Signer l'application

    D'abord créer le keystore : `keytool -genkey -v -keystore nomdelacle.keystore -alias nomdelalias -keyalg RSA -keysize 2048 -validity 10000`

  - `/usr/lib/jvm/java-8-oracle/bin/jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore ~/LAROCHE.keystore platforms/android/app/build/outputs/apk/release/app-release-unsigned.apk P.Laroche`

  - Optimiser l'APK

    `/home/laroche5/android-sdks/build-tools/28.0.3/zipalign -v 4 platforms/android/app/build/outputs/apk/release/app-release-unsigned.apk LeNomDeVotreAppli.apk`

Compléments

- L'entête `Content-Security-Policy`

  dans le header de l'index.html généré, cordova insère la ligne suivante :

```
<meta http-equiv="Content-Security-Policy"
content="default-src 'self' data: gap:
https://ssl.gstatic.com 'unsafe-eval'; style-src
'self' 'unsafe-inline'; media-src *; img-src 'self'
data: content:;">
```

  → cela va empêcher d'inclure des scripts, feuilles de styles, images, appels ajax externes

  → nécessité d'insérer l'url des ressources externes dans cette directive

Compléments

- Toutes les applications cordova utilisent la même icône par défaut

- L'icône à utiliser peut être spécifiée dans config.xml

```
<platform name="android">
    <icon src="res/android/ldpi.png" density="ldpi" />
    <icon src="res/android/mdpi.png" density="mdpi" />
    <icon src="res/android/hdpi.png" density="hdpi" />
    <icon src="res/android/xhdpi.png" density="xhdpi" />
</platform>
```

Application

- Créer une appli cordova, la tester sur browser, émulateur ou smartphone
- Vider le dossier www, puis télécharger l'archive du cours WebMobile et l'extraire dans ce dossier, tester
- Ajouter une page produits.html qui affiche les produits de la catégorie passée en paramètre, grâce au script php indiqué sur Arche
- Déposer sur Arche en fin de séance

Mini-projet

- Création d'une application mobile orientée tourisme
- L'aspect *responsive* smartphone / tablette est à prendre en compte
- A partir d'une base de données de lieux géolocalisés, votre app doit afficher une carte sur laquelle ces lieux sont matérialisés
- La position de l'utilisateur est elle aussi indiquée et remise à jour régulièrement

Mini-projet

- A proximité d'un lieu (moins de 10 mètres) :
  - Le smartphone vibre
  - La carte se réduit
  - Un descriptif avec photos du lieu atteint s'affiche sur environ 3/4 de la page (avec navigation entre pages si la description du lieu le nécessite)
- Autres fonctionnalités possibles :
  - Marquer un lieu comme vu
  - Proposer un itinéraire vers un lieu
  - etc.

Mini-projet

- Les informations touristiques viennent d'une base de données
- Les photos viennent d'un serveur ou d'un base de données
- Technologies à utiliser
  - html/css/js avec ou sans framework js
  - Bootstrap ou équivalent
  - Plugin cordova geolocation
  - Leaflet pour les cartes https://leafletjs.com/examples/mobile/

Mini-projet

- Au niveau des lieux « touristiques », prévoir au moins :
  - IUT
  - Cathédrale
  - Temple protestant
- A réaliser en binômes, pour le 2 février
- A déposer sur Arche : une archive zip contenant toutes les sources, l'apk et éventuellement l'app iOS