

Réaliser des applications Android et IOS  
en HTML/CSS et JavaScript

Greta Sud Aquitaine - 2018

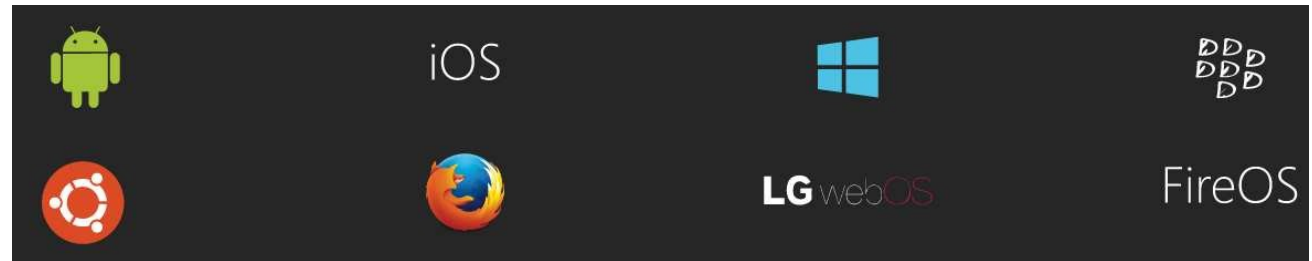
# Introduction au Web sur Mobile

- **Apache Cordova / PhoneGap** : Qu'est-ce que c'est ?
  - Cordova est un pont, un **wrapping** entre JavaScript et le langage natif de l'environnement mobile (**Java** : Android, **Objective-C** : iOS).
  - 1) Il permet de mettre à disposition des développeurs **un ensemble d'APIs qui unifient les appels aux devices natifs** de l'appareil mobile.
  - 2) Il propose un **environnement d'exécution Full Web** (Navigateur natif du système mobile). Par conséquent, le développement s'organise autour d'**HTML5/CSS et JavaScript + des plugins Cordova en JavaScript + Java pour le lanceur de l'application**.
  - 3) Cordova propose également **un outil en ligne de commande** pour organiser son développement :
    - Création d'un template d'application pour démarrer,
    - Génération du package pour la(les) cible(s) (Android AK)
    - Déploiement sur l'appareil mobile (via Android SDK, Xcode)

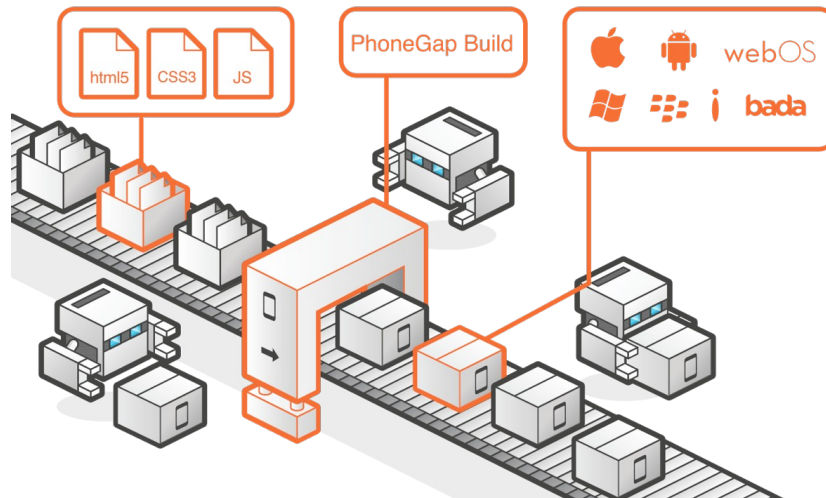
# Introduction au Web sur Mobile

- **Les cibles : Cordova/PhoneGap permet de créer des applications mobiles destinées aux principaux acteurs du marché :**

- iOS
- Android
- Windows Phone (R.I.P.)
- Firefox OS



- **C'est en gros une grosse webview html/css/js dans une app.**

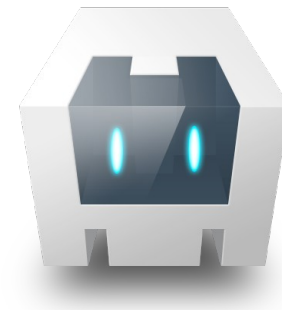
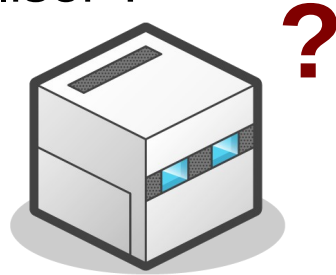


Différence entre Apache Cordova et PhoneGAP :

- **Cordova** est le moteur, les API permettant de Développer les applications.
- **PhoneGAP** est une Distribution de Cordova, il fournis des outils onLine pour simplifier la génération des applications natives.

# Introduction au Web sur Mobile

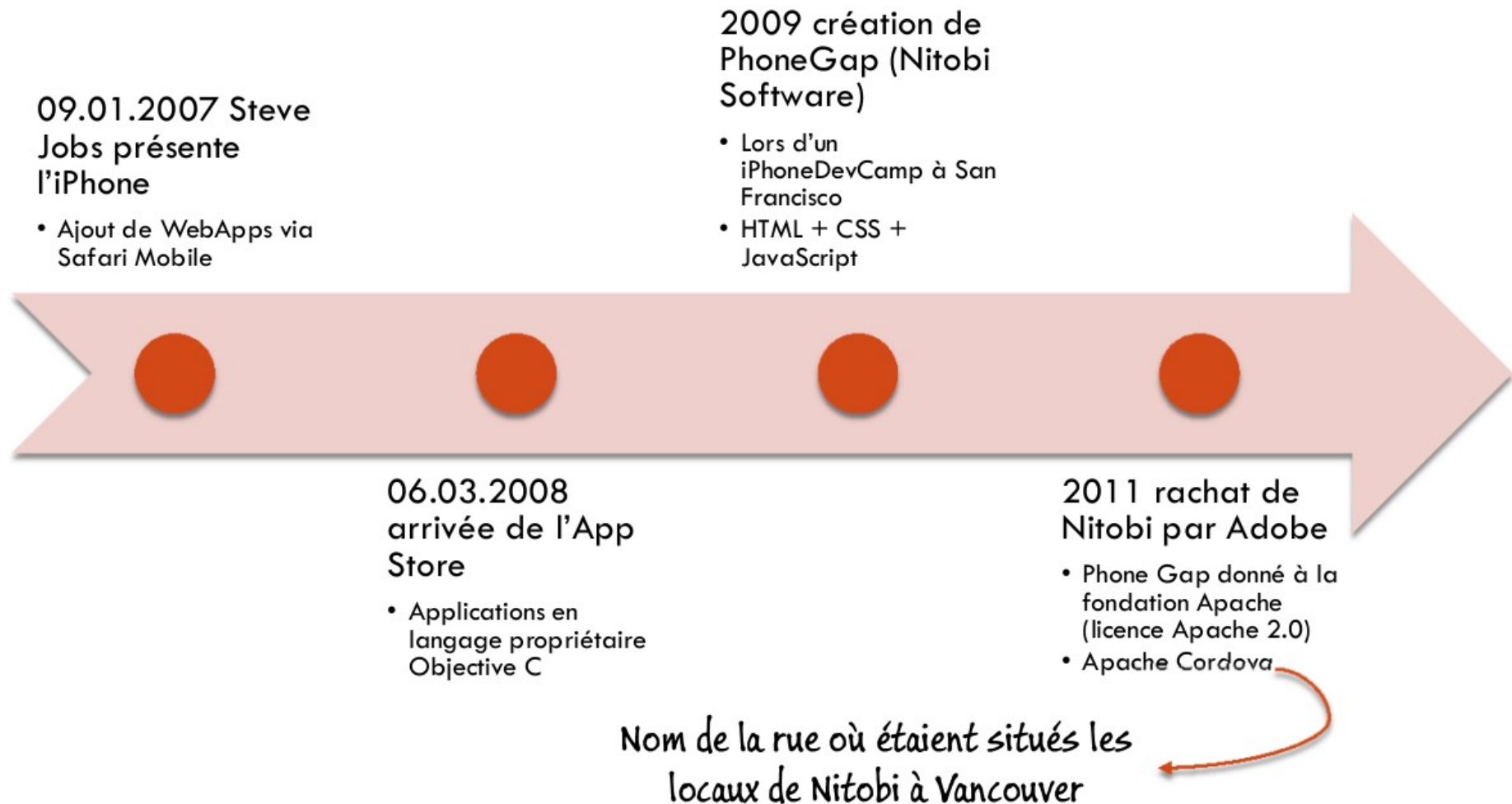
- Lequel utiliser ?



- Pour ce cours **nous allons utiliser Apache Cordova**, car il est plus récent, actuellement en [Version 8.0](#)
- En effet [PhoneGap profite des update de Cordova](#), mais parfois avec un petit décalage.
- Par la suite, libre à vous d'utiliser l'un ou l'autre sachant que c'est le [même framework au départ](#), et vous devrez choisir celui qui vous convient le mieux.
- Avant l'achat par Adobe, [PhoneGap désignait le projet open source](#) et lors de sa reprise par la [fondation Apache](#), le projet devient [Apache Cordova](#)
- Aujourd'hui, [PhoneGap fait référence aux solutions estampillées Adobe](#) utilisant la couche technique Apache Cordova.

# Introduction au Web sur Mobile

- Un peu d'histoire sur cette technologie



- **Pré-requis et logiciels**

- PhoneGap / Cordova se basent sur **des connaissances clients** pour permettre aux développeurs la **construction d'applications mobile** sans avoir de connaissances approfondies en langages natifs.
- Cependant, une **connaissance du développement** natif vous permettra de **développer vos propres plugins** multiplate-forme.
- Avec les langages suivants, le développeur peut très bien développer un projet mobile à destination d'iOS ou d'Android :



Javascript  
HTML (HTML5)  
CSS (CSS3)

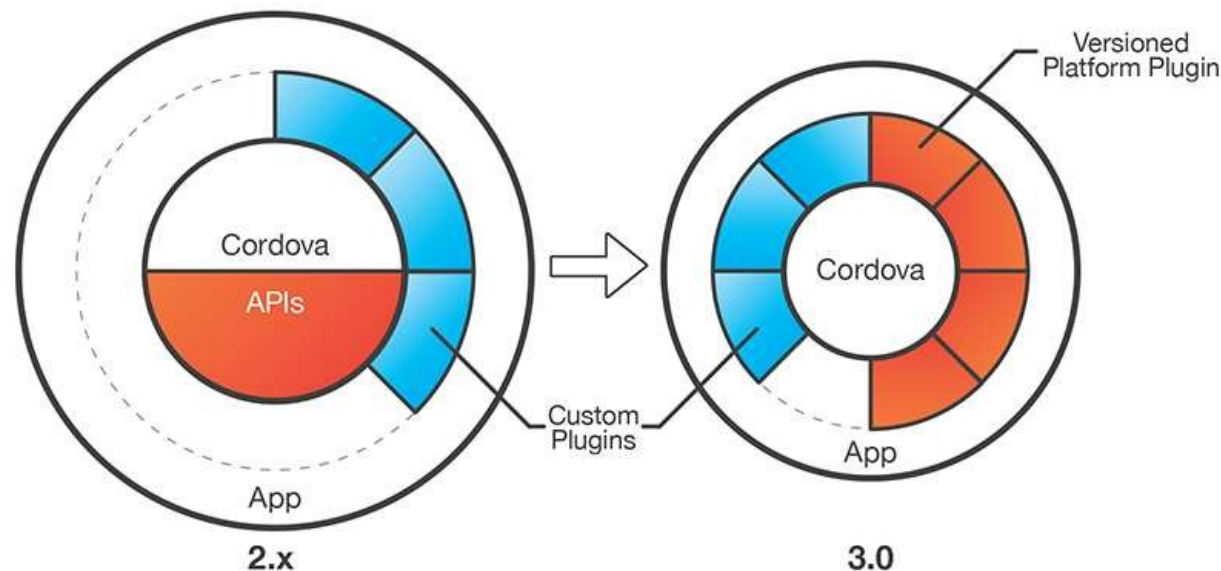


- Le **JavaScript** est une **brique très importante** pour le développement d'applications via PhoneGap / Cordova, car l'intégralité des **manipulations des connecteurs clients -> machine** vont se faire dans ce langage.

# Introduction au Web sur Mobile

- **Architecture de Cordova :**

- Depuis la version 3.0, Cordova s'articule autour d'un modèle « **coeur** » / « **plugins** »



- **API disponibles**

- Ce cours traite principalement d'android, mais **vous pourrez utiliser le code pour iOS, et d'autres environnements mobiles** en fonction de la compatibilité des implémentations du Wrapping Cordova entre le code **JavaScript** ↔ **le code Natif**.

(cf. Tableau des plate-formes supportés ci-après).



# Support des plate-formes

	Amazon-fireos	Android	blackberry10	Firefox OS	iOS	Ubuntu	wp8 (Windows Phone 8)	Windows (8.0, 8.1, 10, Téléphone 8.1)	paciarelli
Cordova CLI	✓ Mac, Windows, Linux	✓ Mac, Windows, Linux	✓ Mac, Windows	✓ Mac, Windows, Linux	✓ Mac	✓ Ubuntu	✓ Windows	✓	✗
Embedded WebView	✓ (voir détails)	✓ (voir détails)	✗	✗	✓ (voir détails)	✓	✗	✗	✗
Plugin Interface	✓ (voir détails)	✓ (voir détails)	✓ (voir détails)	✗	✓ (voir détails)	✓	✓ (voir détails)	✓	✗
API de la plate-forme									
Accéléromètre	✓	✓	✓	✓	✓	✓	✓	✓	✓
BatteryStatus	✓	✓	✓	✓	✓	✗	✓	✓ * Windows Phone 8.1 seulement	✓
Appareil photo	✓	✓	✓	✓	✓	✓	✓	✓	✓
Capture	✓	✓	✓	✗	✓	✓	✓	✓	✗
Boussole	✓	✓	✓	✗	✓ (3 G +)	✓	✓	✓	✓
Connexion	✓	✓	✓	✗	✓	✓	✓	✓	✓
Contacts	✓	✓	✓	✓	✓	✓	✓	partiellement	✗
Dispositif	✓	✓	✓	✓	✓	✓	✓	✓	✓
Événements	✓	✓	✓	✗	✓	✓	✓	✓	✓
Fichier	✓	✓	✓	✗	✓	✓	✓	✓	✗
Transfert de fichiers	✓	✓	✓ * Ne pas soutenir onprogress ni abandonner.	✗	✓	✗	✓ * Ne pas soutenir onprogress ni abandonner.	✓ * Ne pas soutenir onprogress ni abandonner.	✗
Géolocalisation	✓	✓	✓	✓	✓	✓	✓	✓	✓
Mondialisation	✓	✓	✓	✗	✓	✓	✓	✓	✗
InAppBrowser	✓	✓	✓	✗	✓	✓	✓	utilise les iframe	✗
Media	✓	✓	✓	✗	✓	✓	✓	✓	✓
Notification	✓	✓	✓	✗	✓	✓	✓	✓	✓
SplashScreen	✓	✓	✓	✗	✓	✓	✓	✓	✗
Barre d'État	✗	✓	✗	✗	✓	✗	✓	✓ 8.1 de Windows Phone uniquement	✗
Stockage	✓	✓	✓	✗	✓	✓	✓ localStorage & indexedDB	✓ localStorage & indexedDB	✓
Vibration	✓	✓	✓	✓	✓	✗	✓	✓ * Windows Phone 8.1 seulement	✗



# Introduction au Web sur Mobile

- **Installation de l'environnement de développement :**
  - Prérequis :
    - Vous devez avoir un « **JDK** » d'installer sur votre système, ainsi que « **Ant** ».

```
jf@uuser1604:~/android-sdk-linux/tools$ java -version
openjdk version "1.8.0_91"
```

- Avant d'utiliser Cordova CLI, en ligne de commande, vous devez installer le SDK correspondant à chaque plate-forme ciblée.
  - **Android SDK** : Ouvrir cette page :  
<http://developer.android.com/studio/index.html#downloads>

```
jf@uuser1604:~$ wget https://dl.google.com/android/android-sdk_r24.4.1-linux.tgz
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : 326412652 (311M) [application/x-tar]
Enregistre : «android-sdk_r24.4.1-linux.tgz»
~$ tar zxvf android-sdk_r24.4.1-linux.tgz
(...)
```

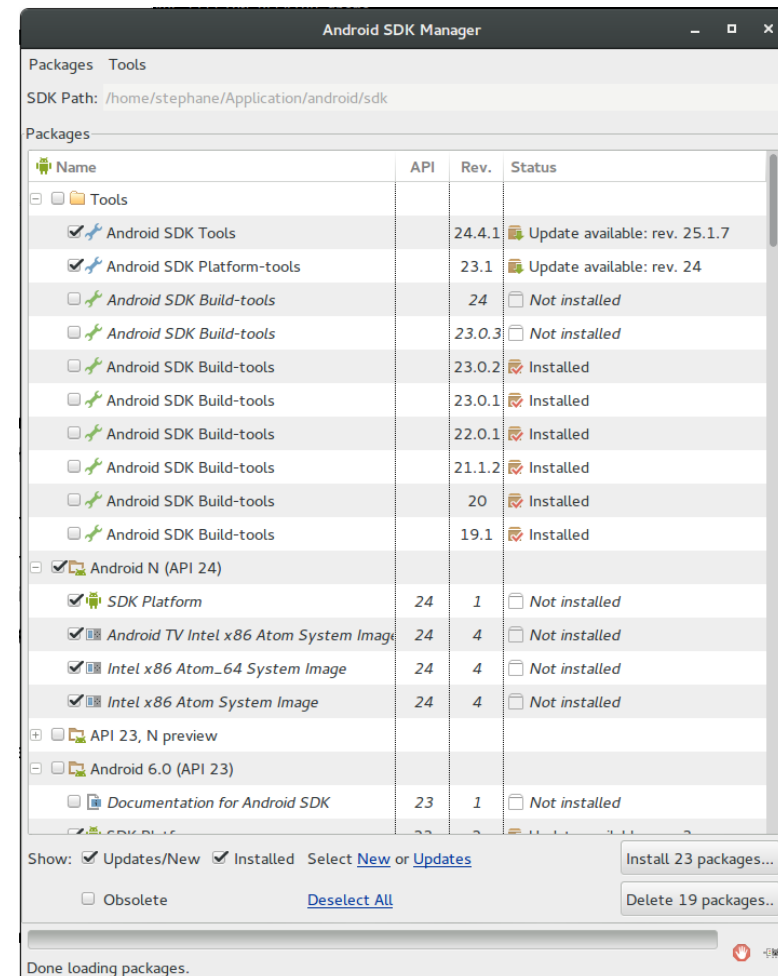
```
~$ export PATH=$PATH:/home/stephane/development/android-sdk-linux/tools/tools
~$ export PATH=$PATH:/home/stephane/development/android-sdk-linux/platform-tools
stephane@UX303UB:~/ $ android
```

- Vous devez lancer l'exécutable « **android** » pour choisir votre SDK :

# Android SDK Manager

- « **Android SDK Manager** » est une application graphique qui permet de charger et mettre à jour les différents SDK en fonction de leur version d'API.
- Si vous avez déjà développé pour android des applications Natives, il doit être installé.
- Il faut ajouter le dossier « **platform-tools** » au PATH après avoir installé **SDK Platform-tools**.

*NB : Pour mettre à jour le sdk sur un serveur (machine virtuelle) il existe une option en mode **non-ui**.*



```
jf@uuser1604:~/cordova/platform-tools$ ./android update sdk --no-ui
```

```
export ANDROID_HOME="/home/jf/android-sdk-linux/"
export PATH="$ANDROID_HOME/tools:$PATH"
export PATH="$ANDROID_HOME/platform-tools:$PATH"
```

Extrait du fichier  
« .bashrc » de la  
machine virtuelle)

# Apache Cordova CLI

- L'outil de « **Command Line Interface** » d'Apache Cordova s'installe de façon très simple comme un module Node.js :
  - Une **check liste des pré-requis s'impose** avant de lancer l'installation de Cordova CLI, utilisez la succession des commandes suivantes :

```
jf@uuser1604:~/android-sdk-linux/tools$ ll
total 32
drwxrwxr-x  7 jf jf 4096 juil.  2 21:19 ./
drwxr-xr-x  7 jf jf 4096 juil.  2 17:08 ../
drwxrwxr-x  2 jf jf 4096 oct.  14  2015 add-ons/
drwxrwxr-x  8 jf jf 4096 juil.  2 21:20 build-tools/
drwxrwxr-x  6 jf jf 4096 juil.  2 17:32 platforms/
drwxrwxr-x  5 jf jf 4096 juil.  2 18:15 platform-tools/
-rw-rw-r--  1 jf jf 1158 oct.  14  2015 SDK Readme.txt
drwxr-xr-x 12 jf jf 4096 oct.  14  2015 tools/
jf@uuser1604:~/android-sdk-linux/tools$ java -version
openjdk version "1.8.0_91"
OpenJDK Runtime Environment (build 1.8.0_91-8u91-b14-0ubuntu4~16.04.1-b14)
OpenJDK 64-Bit Server VM (build 25.91-b14, mixed mode)
jf@uuser1604:~/android-sdk-linux/tools$ ant -version
Apache Ant(TM) version 1.9.6 compiled on July 8 2015
jf@uuser1604:~/android-sdk-linux/tools$ node -v
v4.2.6
jf@uuser1604:~/android-sdk-linux/tools$ npm -v
3.5.2
```

- Si vous avez les mêmes réponses sans erreur au numéro de version prêt, la suite !

# Apache Cordova CLI

- **Installons Cordova CLI, enfin ;-)**
  - Comme on installe un module Node.js avec npm et l'option « -g »

```
stephane@UX303UB:~$ npm install -g cordova
npm WARN deprecated node-uuid@1.4.8: Use uuid module instead
/home/stephane/.nvm/versions/node/v5.0.0/bin/bin/cordova ->
/home/stephane/.nvm/versions/node/v5.0.0/bin/lib/node_modules/cordova/bin/cordova
cordova@6.5.0 /home/stephane/.nvm/versions/node/v5.0.0/bin/lib/node_modules/cordova
├── underscore@1.7.0
├── q@1.0.1
├── nopt@3.0.1 (abbrev@1.1.0)
├── update-notifier@0.5.0 (is-npm@1.0.0, semver-diff@2.1.0, string-length@1.0.1,
  │ repeating@1.1.3, chalk@1.1.3, configstore@1.4.0,
  │ latest-version@1.0.1)
├── insight@0.8.4 (object-assign@4.1.1, async@1.5.2, uuid@3.0.1, lodash.debounce@3.1.1,
  │ tough-cookie@2.3.2, chalk@1.1.3, os-name@1.0.3, configstore@1.4.0, request@2.81.0,
  │ inquirer@0.10.1)
├── cordova-common@2.0.0 (cordova-registry-mapper@1.1.15, unorm@1.4.1, underscore@1.8.3,
  │ q@1.5.0, semver@5.3.0, ansi@0.3.1, osenv@0.1.4, bplist-parser@0.1.1, minimatch@3.0.3,
  │ glob@5.0.15, elementtree@0.1.7, shelljs@0.5.3, plist@1.2.0)
├── cordova-lib@6.5.0 (valid-identifier@0.0.1, opener@1.4.1, cordova-registry-mapper@1.1.15,
  │ unorm@1.3.3, properties-parser@0.2.3, nopt@3.0.6, semver@4.3.6, dep-graph@1.1.0,
  │ shelljs@0.3.0, glob@5.0.15, elementtree@0.1.6, request@2.47.0, xcode@0.9.3,
  │ cordova-serve@1.0.1, aliasify@1.9.0, tar@1.0.2, init-package-json@1.9.5,
  │ cordova-fetch@1.0.2, cordova-create@1.0.2, plist@1.2.0, npm@2.15.12, cordova-js@4.2.1)
```

- **Installation d'un CVM Cordova Version Manager :**  
<https://www.npmjs.com/package/version-manager-cordova-software>
- **Vérification de l'installation**
  - Ensuite on peut vérifier en tapant « **cordova** » en ligne de commande et voir l'ensemble des outils mis à disposition.

# Apache Cordova CLI

- **Notre première application Apache Cordova**

- Si ce n'est pas fait créé un dossier « **cordova** » dans lequel nous mettrons les projets.
- Tapez la commande suivante pour créer un template d'application Apache Cordova

```
jf@uuser1604:~/cordova$ cordova create HelloWorld com.example.hello "Hello World"
Creating a new cordova project.
jf@uuser1604:~/cordova$ ll
total 12
drwxrwxr-x  3 jf jf 4096 juil.  3 02:13 ./
drwxr-xr-x 11 jf jf 4096 juil.  3 02:13 ../
drwxrwxr-x  6 jf jf 4096 juil.  3 02:13 HelloWorld/
jf@uuser1604:~/cordova$ cd HelloWorld/
jf@uuser1604:~/cordova/HelloWorld$ ll
total 28
drwxrwxr-x 6 jf jf 4096 juil.  3 02:13 ./
drwxrwxr-x 3 jf jf 4096 juil.  3 02:13 ../
-rw-r--r-- 1 jf jf  985 juil.  3 02:13 config.xml
drwxrwxr-x 2 jf jf 4096 juil.  3 02:13 hooks/
drwxrwxr-x 2 jf jf 4096 juil.  3 02:13 platforms/
drwxrwxr-x 2 jf jf 4096 juil.  3 02:13 plugins/
drwxrwxr-x 5 jf jf 4096 juil.  3 02:13 www/
```

*NB : le dossier « **platforms** » est vide nous devons ajouter les environnements mobiles*

# Apache Cordova CLI

- Pour ajouter une plate-forme cible à notre application, nous allons utiliser la commande suivante :

```
jf@uuser1604:~/cordova/HelloWorld$ cordova platform add android
Adding android project...
Creating Cordova project for the Android platform:
  Path: platforms/android
  Package: com.example.hello
  Name: Hello_World
  Activity: MainActivity
  Android target: android-23
Android project created with cordova-android@5.1.1
Discovered plugin "cordova-plugin-whitelist" in config.xml. Adding it to the project
Fetching plugin "cordova-plugin-whitelist@1" via npm
Installing "cordova-plugin-whitelist" for android
```

- On peut vérifier les plate-formes cibles installées dans une application avec la commande suivante, toujours dans le dossier de l'application

```
jf@uuser1604:~/cordova/HelloWorld/platforms$ cordova platform list
Installed platforms:
  android 5.1.1
Available platforms:
  amazon-fireos ~3.6.3 (deprecated)
  blackberry10 ~3.8.0
  browser ~4.1.0
  firefoxos ~3.6.3
  ubuntu ~4.3.3
  webos ~3.7.0
```

# Apache Cordova CLI

- Il est donc normalement possible, si vous avez tout configuré, de lancer une compilation et une génération de notre application

```
jf@uuser1604:~/cordova/HelloWorld$ cordova build android
ANDROID_HOME=/home/jf/android-sdk-linux/
JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
Downloading http://services.gradle.org/distributions/gradle-2.2.1-all.zip
.....
BUILD SUCCESSFUL

Total time: 24.21 secs
Built the following apk(s):
  /home/jf/cordova/HelloWorld/platforms/android/build/outputs/apk/android-debug.apk
```

- Il se peut que vous rencontriez des problèmes de compilation, lors de l'installation sur la machine virtuelle de formation, une erreur est apparue, pour Ubuntu 16.04 serveur (mode --no-ui) .

```
jf@uuser1604:~/cordova/HelloWorld$ cordova build android
ANDROID_HOME=/home/jf/android-sdk-linux/
JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
(...)
:CordovaLib:processDebugResources FAILED
FAILURE: Build failed with an exception.
* What went wrong:
Execution failed for task ':CordovaLib:processDebugResources'.
```

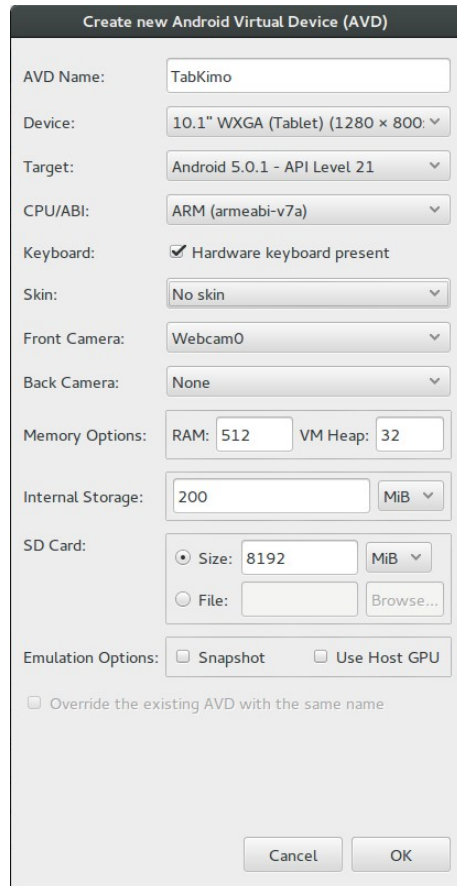
- Solution : installer 2 librairie C++.

```
jf@uuser1604:~/cordova/HelloWorld$ sudo apt-get install lib32stdc++6 lib32z1
```



# Machine virtuelle Android

- Pour configurer une machine virtuelle qui pourra être utilisée par défaut il suffit d'utiliser le SDK Android, dans le menu « Tools » choisir « **Manage AVDs...** »
- Vous devez, créer un type de machine virtuel au plus prêt de votre matériel cliquez sur « **create** »



AVD Name: TabKimo

Device: 10.1" WXGA (Tablet) (1280 x 800)

Target: Android 5.0.1 - API Level 21

CPU/ABI: ARM (armeabi-v7a)

Keyboard: ☒ Hardware keyboard present

Skin: No skin

Front Camera: Webcam0

Back Camera: None

Memory Options: RAM: 512 VM Heap: 32

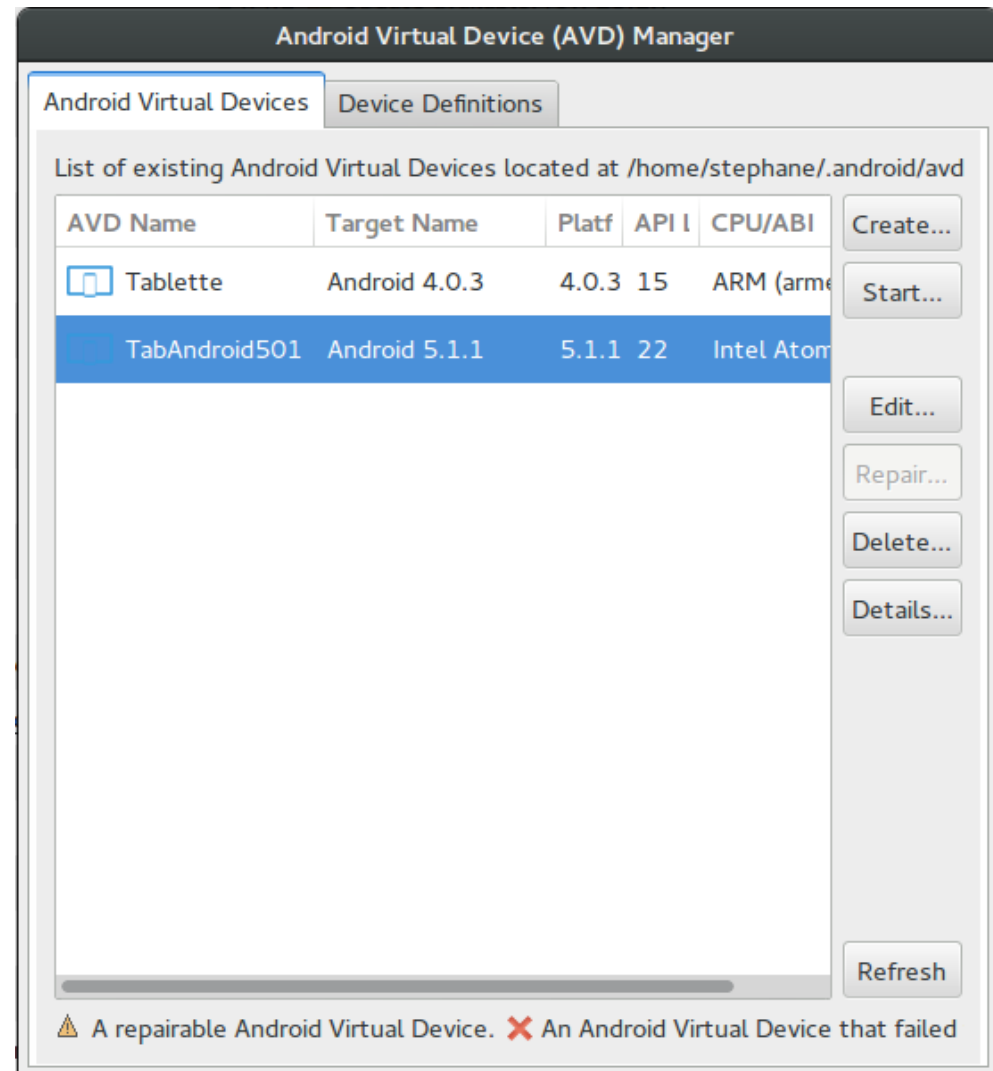
Internal Storage: 200 MiB

SD Card: ☒ Size: 8192 MiB ☐ File: Browse...

Emulation Options: ☐ Snapshot ☐ Use Host GPU

☐ Override the existing AVD with the same name

Cancel OK

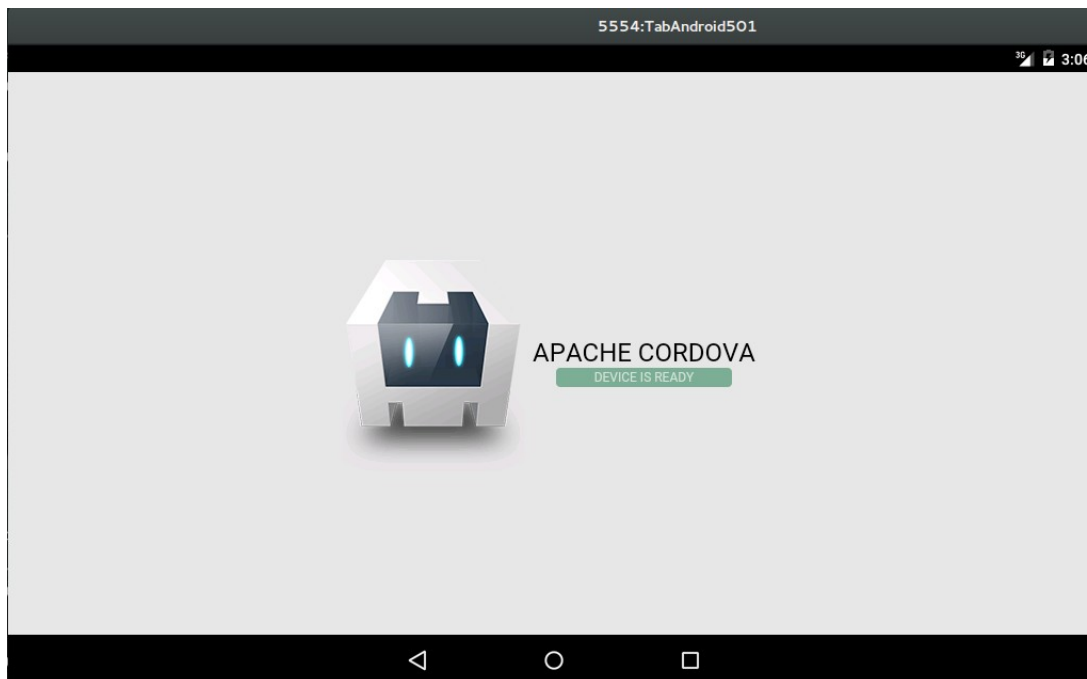




# Apache Cordova CLI

- Dans un environnement graphique, il est possible de lancer l'application via Cordova CLI :

```
stephane@UX303UB:~/formation/Clients/Kimo/TP/HelloWorld$ cordova run android
ANDROID_HOME=/home/stephane/Application/android/sdk
JAVA_HOME=/usr/lib/jvm/java-7-oracle/
No target specified, deploying to emulator
No emulator specified, defaulting to TabAndroid501
Waiting for emulator...
emulator: UpdateCheck: current version '24.4.1', last version '24.4.1'
Booting up emulator (this may take a while)...BOOT COMPLETE
(...)
```

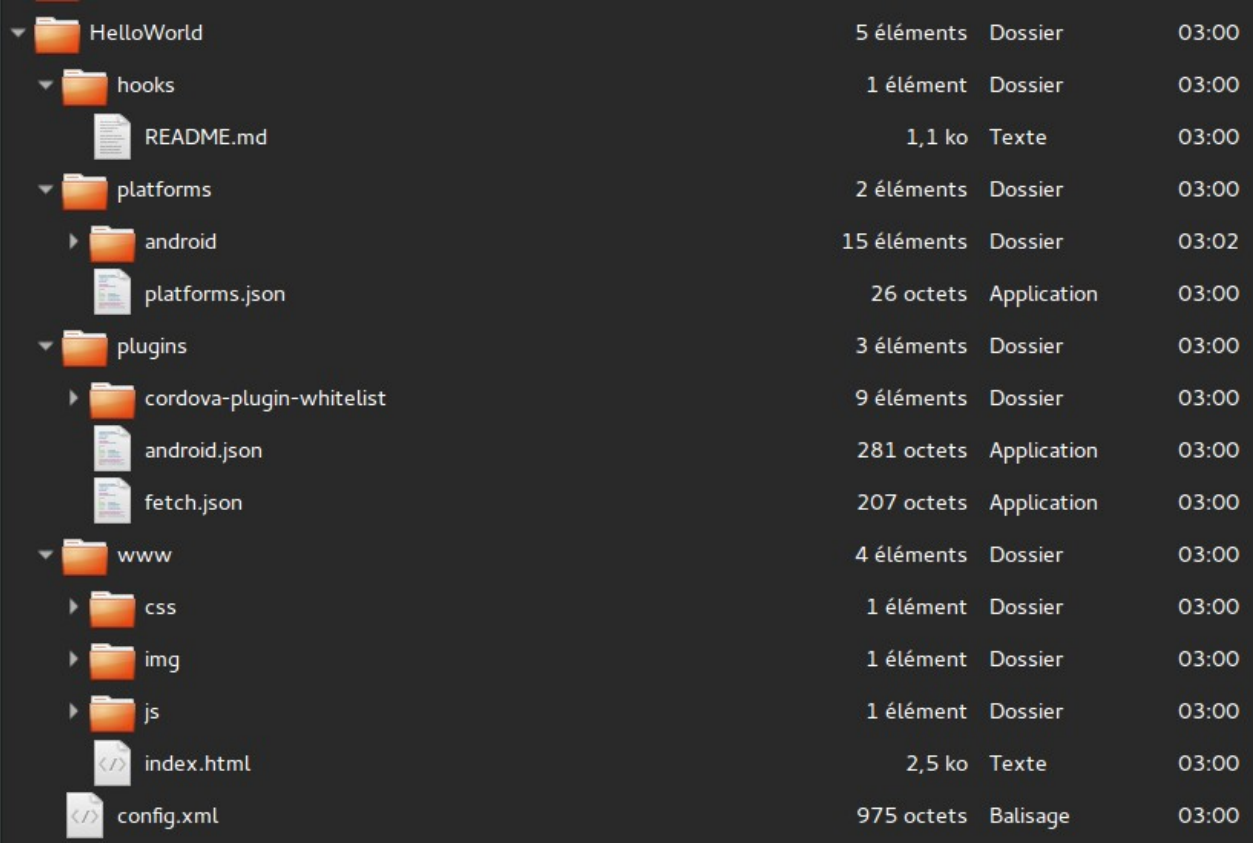


- Si vous aviez connecté votre tablette sur le port USB de l'ordinateur qui contient le code de l'application Cordova. Alors avec une commande de type « **run** » l'application s'installe sur le device physique (par défaut) et virtuel si pas de device physique trouvé.

# Analyse du code Cordova généré

- Nous allons voir les codes sources de l'application « [HelloWorld](#) » pour comprendre comment cela fonctionne et comment réutiliser cette génération de code pour commencer notre application.

– L'arborescence :



▼ HelloWorld	5 éléments	Dossier	03:00
▼ hooks	1 élément	Dossier	03:00
README.md	1,1 ko	Texte	03:00
▼ platforms	2 éléments	Dossier	03:00
▶ android	15 éléments	Dossier	03:02
platforms.json	26 octets	Application	03:00
▼ plugins	3 éléments	Dossier	03:00
▶ cordova-plugin-whitelist	9 éléments	Dossier	03:00
android.json	281 octets	Application	03:00
fetch.json	207 octets	Application	03:00
▼ www	4 éléments	Dossier	03:00
▶ css	1 élément	Dossier	03:00
▶ img	1 élément	Dossier	03:00
▶ js	1 élément	Dossier	03:00
index.html	2,5 ko	Texte	03:00
config.xml	975 octets	Balisateur	03:00

- Le fichier de configuration « [config.xml](#) » est important c'est lui qui va décrire les options de l'application native. Vous trouverez le code de l'application dans le dossier « [www](#) ».

# Analyse du code Cordova généré

- Voici le code HTML : « [index.html](#) »

```
<html>
  <head>
    <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 *">
    <meta name="format-detection" content="telephone=no">
    <meta name="msapplication-tap-highlight" content="no">
    <meta name="viewport" content="user-scalable=no, initial-scale=1,
      maximum-scale=1, minimum-scale=1, width=device-width">
    <link rel="stylesheet" type="text/css" href="css/index.css">
    <title>Hello World</title>
  </head>
  <body>
    <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>
  </body>
</html>
```

Le fichier « [cordova.js](#) » se trouve dans « [HelloWorld/platforms/android/assets/www](#) »

# Analyse du code Cordova généré



- Voici le code JavaScript applicatif : « [index.js](#) »

```
var app = {  
  // Application Constructor  
  initialize: function() {  
    this.bindEvents();  
  },  
  // Bind Event Listeners, bind any events that are required on startup.  
  // Common events are: 'load', 'deviceready', 'offline', and 'online'.  
  bindEvents: function() {  
    document.addEventListener('deviceready', this.onDeviceReady, false);  
  },  
  // deviceready Event Handler  
  // The scope of 'this' is the event. In order to call the 'receivedEvent'  
  // function, we must explicitly call 'app.receivedEvent(...)';  
  onDeviceReady: function() {  
    app.receivedEvent('deviceready');  
  },  
  // Update DOM on a Received Event  
  receivedEvent: function(id) {  
    var parentElement = document.getElementById(id);  
    var listeningElement = parentElement.querySelector('.listening');  
    var receivedElement = parentElement.querySelector('.received');  
    listeningElement.setAttribute('style', 'display:none;');  
    receivedElement.setAttribute('style', 'display:block;');  
    console.log('Received Event: ' + id);  
  }  
};  
app.initialize();
```

# Analyse du code Cordova généré



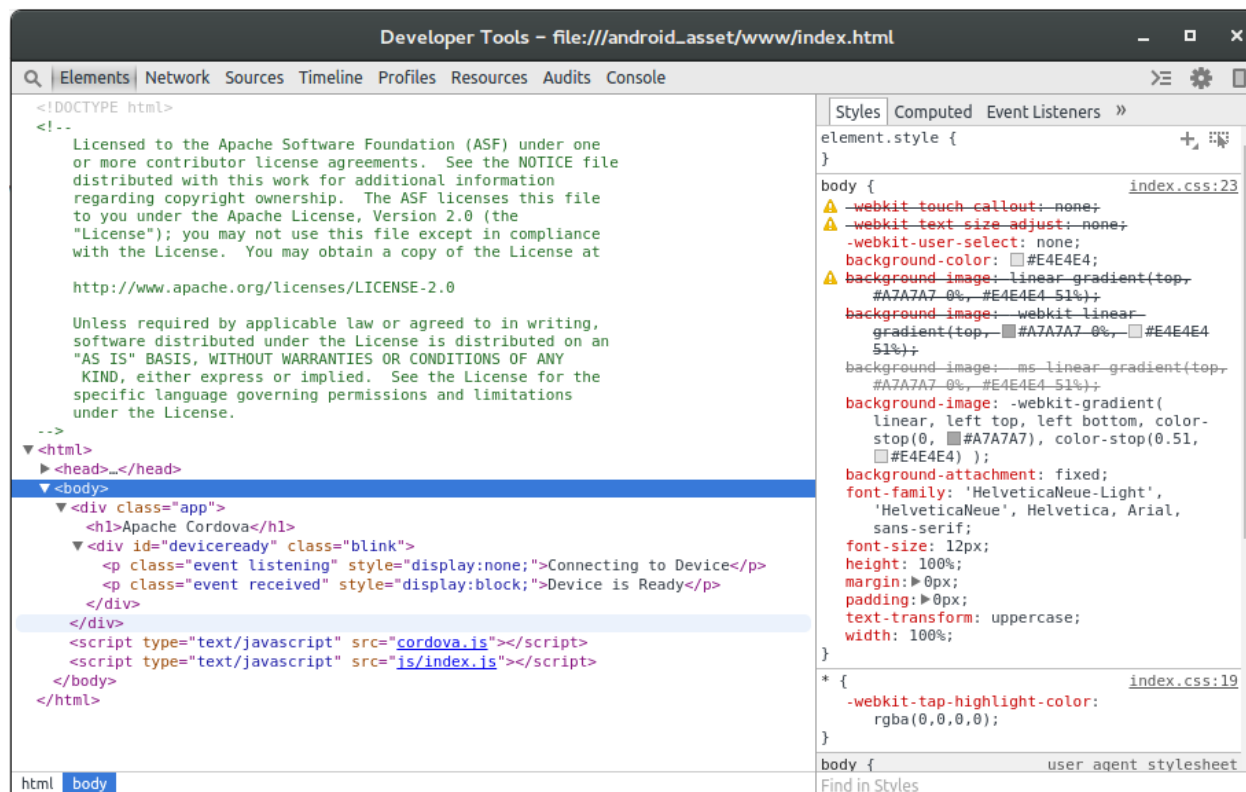
- Le fichier « [config.xml](#) » (description [page 54](#) du Livre « [Application Mobile avec Cordova...](#) »)

```
<?xml version='1.0' encoding='utf-8'?>
<widget id="fr.kimo" version="0.0.1" xmlns="http://www.w3.org/ns/widgets"
  xmlns:cdv="http://cordova.apache.org/ns/1.0">
  <name>Hello World</name>
  <description>
    A sample Apache Cordova application that responds to the deviceready event.
  </description>
  <author email="dev@cordova.apache.org" href="http://cordova.io">
    Apache Cordova Team
  </author>
  <content src="index.html" />
  <plugin name="cordova-plugin-whitelist" spec="1" />
  <access origin="*" />
  <allow-intent href="http://*/*" />
  <allow-intent href="https://*/*" />
  <allow-intent href="tel:*" />
  <allow-intent href="sms:*" />
  <allow-intent href="mailto:*" />
  <allow-intent href="geo:*" />
  <platform name="android">
    <allow-intent href="market:*" />
  </platform>
  <platform name="ios">
    <allow-intent href="itms:*" />
    <allow-intent href="itms-apps:*" />
  </platform>
</widget>
```

# Debuguer l'App Cordova généré

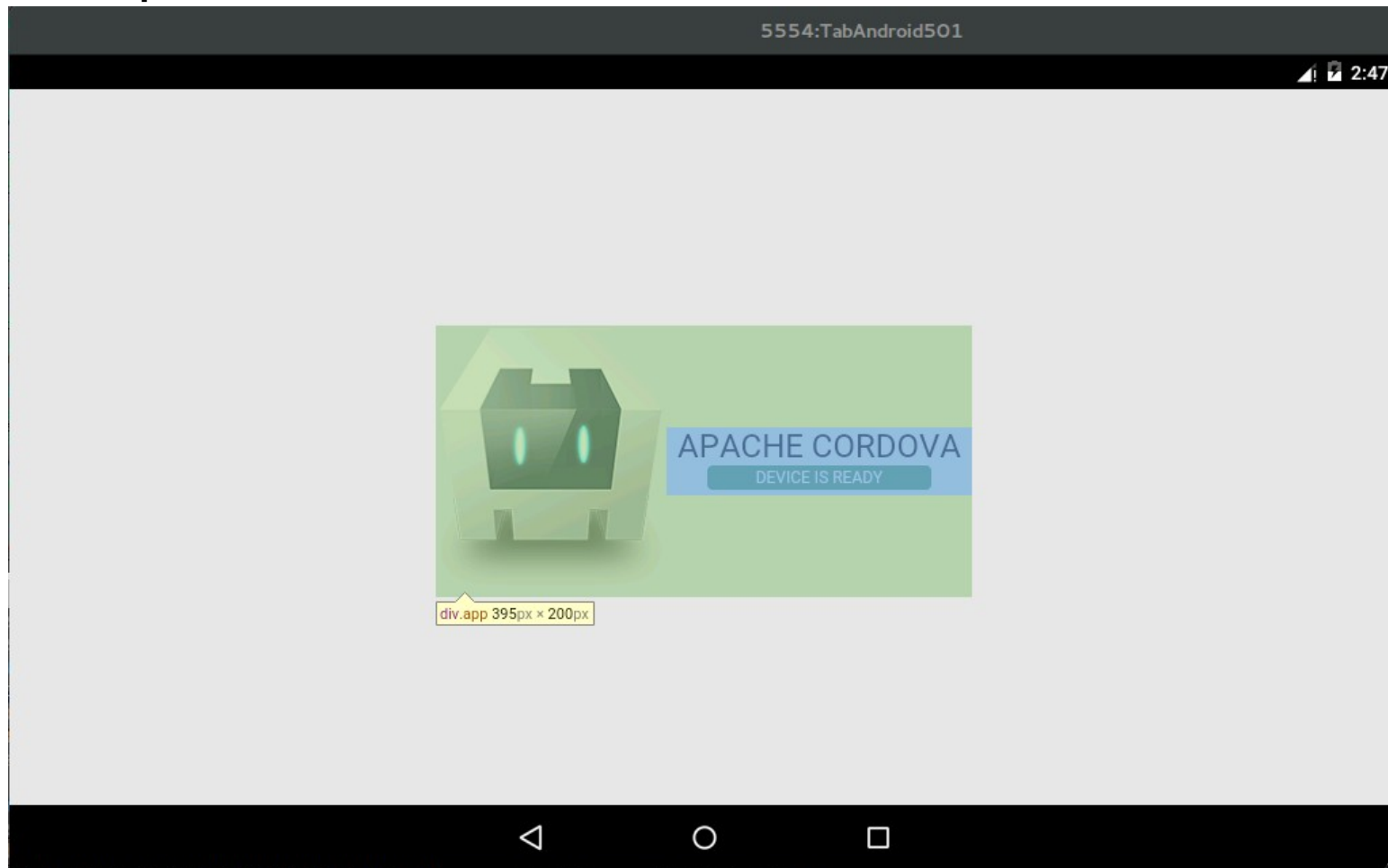
- Afin de comprendre les enchainements lors de l'exécution de l'application, il est possible de debuguer l'application, soit lorsque l'on a un **Device** physique connecté, soit lorsque l'on a l'Emulateur en cours d'exécution après un « **cordova run android** ».
  - Pour ce faire il suffit de lancer un chrome ou chromium-browser et de taper l'URL suivante dans la barre d'adresse du navigateur :

**chrome://inspect**



# Debuguer l'App Cordova généré

- Voici la vue dans l'Emulateur en fonction des bloc HTML que l'on sélectionne l'outils de développement de chrome affiche des informations. Il est aussi possible de debuguer en Javascript.

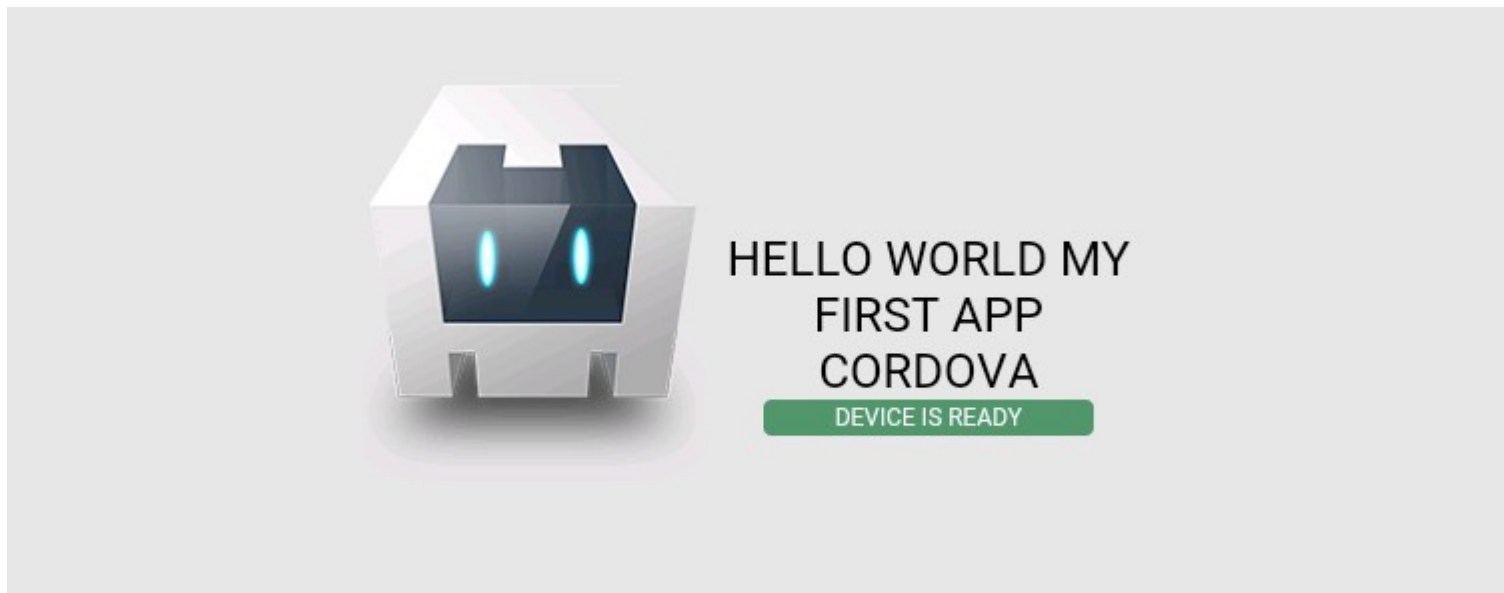


# Debuguer l'App Cordova généré

- Lorsque vous faites une modification sur votre code source, pour l'exemple nous allons modifier « APACHE CORDOVA » par « HELLO WORLD MY FIRST APP CORDOVA ».
- Par la suite, il n'est pas nécessaire de fermer la machine virtuelle pour relancer le déploiement, un simple :

```
stephane@UX303UB:~Documents/formation/Client/Kimo/TP/HelloWorld$ cordova run
```

- Cela évite de relancer la machine virtuelle et fait gagner du temps lors du debuggage





# Cordova CLI : Ajouter un plugin

- Vous pouvez ajouter des plugins à votre application.
  - Une liste non exhaustive de quelques plugins principaux
    - White List
    - Crosswalk (plus nécessaire depuis android 4.4)
    - Camera
    - Media Capture
    - DatePicker
    - ...
  - Vous pourrez trouver une liste de plugins en fonction de votre système hôte de votre application à l'adresse suivante :  
<https://cordova.apache.org/plugins/>
  - Actuellement le nombre de plugin disponibles en commun pour les 2 principales plate-formes du marché : Android et iOS est de 702.

# Cordova CLI : Ajouter un plugin

- Nous allons ajouter un plugin pour voir toute la procédure ;-)
  - Les principaux plugins que l'on ajoute en général dans une application sont :
    - cordova plugin add [cordova-plugin-device](#)
    - cordova plugin add [cordova-plugin-network-information](#)
    - cordova plugin add [cordova-plugin-battery-status](#)
    - cordova plugin add [cordova-plugin-device-motion](#)
    - cordova plugin add [cordova-plugin-device-orientation](#)
  - Nous allons installer un outil nous permettant de savoir si notre device est connecté au réseau, il faut utiliser :

```
stephane@UX303UB:~/workspaceGreta/HelloWorld$ cordova plugin add cordova-plugin-network-information
Fetching plugin "cordova-plugin-network-information@~1.2.0" via npm
Installing "cordova-plugin-network-information" for android
```

```
stephane@UX303UB:~/workspaceGreta/HelloWorld/plugins$ ll
total 24
drwxrwxr-x 4 stephane stephane 4096 avril 11 00:00 ./
drwxrwxr-x 6 stephane stephane 4096 avril 10 09:34 ../
-rw-rw-r-- 1 stephane stephane 398 avril 11 00:00 android.json
drwxrwxr-x 7 stephane stephane 4096 avril 11 00:00 cordova-plugin-network-information/
drwxrwxr-x 4 stephane stephane 4096 avril 10 09:45 cordova-plugin-whitelist/
-rw-rw-r-- 1 stephane stephane 430 avril 11 00:00 fetch.json
```

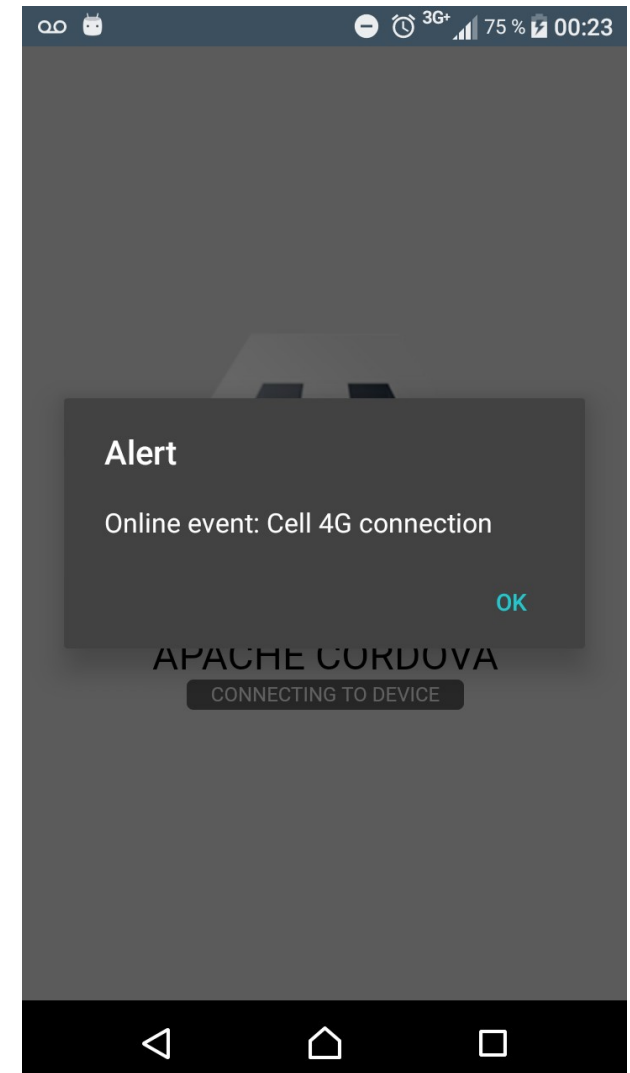
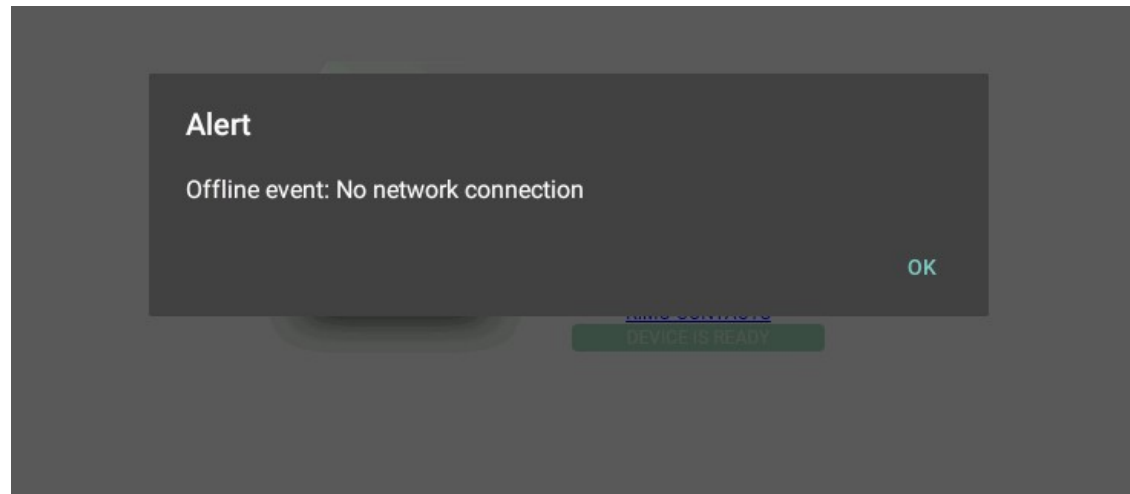
# Cordova CLI : Ajouter un plugin

- Mise en œuvre de notre plugin, pour ce faire toujours regarder la documentation sur le compte github du plugin. Modifions « **index.js** »

```
(...)  
bindEvents: function() {  
    document.addEventListener('deviceready', this.onDeviceReady, false);  
    document.addEventListener("offline", this.onOffline, false);  
    document.addEventListener("online", this.onOnline, false);  
}  
onOffline: function() {  
    var typecnx = checkConnection();  
    alert('Offline event: ' + typecnx);  
},  
onOnline: function() {  
    var typecnx = checkConnection();  
    alert('Offline event: ' + typecnx);  
}  
};  
function checkConnection() {  
    var networkState = navigator.connection.type;  
    var states = {};  
    states[Connection.UNKNOWN] = 'Unknown connection';  
    states[Connection.ETHERNET] = 'Ethernet connection';  
    states[Connection.WIFI] = 'WiFi connection';  
    states[Connection.CELL_2G] = 'Cell 2G connection';  
    states[Connection.CELL_3G] = 'Cell 3G connection';  
    states[Connection.CELL_4G] = 'Cell 4G connection';  
    states[Connection.CELL] = 'Cell generic connection';  
    states[Connection.NONE] = 'No network connection';  
  
    return states[networkState];  
}  
app.initialize() ;
```

# Cordova CLI : Ajouter un plugin

- Le résultat s'affiche dans une alerte et vous donne le type de connexion réseau du Device physique.



# Cordova CLI : Ajouter un plugin

- Fichier contact.html

```
(...)  
bindEvents: function() {  
    document.addEventListener('deviceready', this.onDeviceReady, false);  
    document.addEventListener("offline", this.onOffline, false);  
    document.addEventListener("online", this.onOnline, false);  
}  
onOffline: function() {  
    var typecnx = checkConnection();  
    alert('Offline event: ' + typecnx);  
},  
onOnline: function() {  
    var typecnx = checkConnection();  
    alert('Offline event: ' + typecnx);  
}  
};  
function checkConnection() {  
    var networkState = navigator.connection.type;  
    var states = {};  
    states[Connection.UNKNOWN] = 'Unknown connection';  
    states[Connection.ETHERNET] = 'Ethernet connection';  
    states[Connection.WIFI] = 'WiFi connection';  
    states[Connection.CELL_2G] = 'Cell 2G connection';  
    states[Connection.CELL_3G] = 'Cell 3G connection';  
    states[Connection.CELL_4G] = 'Cell 4G connection';  
    states[Connection.CELL] = 'Cell generic connection';  
    states[Connection.NONE] = 'No network connection';  
  
    return states[networkState];  
}  
app.initialize() ;
```

# Cordova : le code Natif

- Poir pouvoir exécuter votre code HTML5/CSS et JavaScript cordova initialise une classe Java qui lance une webview.
  - On retrouve le code source dans le dossier [platform/android/src](#)

```
package com.exemple.hello;

import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends CordovaActivity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);

        // enable Cordova apps to be started in the background
        Bundle extras = getIntent().getExtras();
        if (extras != null && extras.getBoolean("cdvStartInBackground", false)) {
            moveTaskToBack(true);
        }

        // Set by <content src="index.html" /> in config.xml
        loadUrl(launchUrl);
    }
}
```

**NB : essayez de changer le « launchUrl » en mettant une URL http://...**

# TP

## Ecrire une application cordova avec un formulaire de contact et l'intégrer dans un nouveau projet nommé «**contacts** »

- L'objectif premier est de vous familiariser avec la commandes de génération d'application Cordova.
- De naviguer dans et de modifier le code source généré par la création d'application de Cordova CLI.
- De déployer et visualiser le résultat dans un émulateur et/ou un Device physique (votre téléphone).
- L'application devra permettre de gérer des contacts téléphoniques donc : **nom, prénom, adress1, adresse2, code\_postal, ville, téléphone mobile, téléphone fixe.**
- Vous essaieriez de trouver par vous même une façon de stocker les contacts : vous pouvez utiliser un fichier texte simple (json), une base de données WebSQL ou LocalStorage pour la persistance des contacts.