silexlabs.org et pixaline.net présente…

Haxetelier #10 : Cordova avec Haxe
-- Résumé --

- ## Téléchargement de l'application finale depuis Google Play :

  L'app s'appelle "Apix CordovHaxe" ;
  aller sur https://play.google.com/store/apps/details?id=net.apixline.cordosample
  ou
  taper "apix" dans la recherche.

  Il y a aussi l'application de démo "Apix UICompo" qui contient quelques "plugins" Cordova comme "Camera" ou "Geolocation"

- ## Installations des dépendances

  (NodeJs-Cordova-Java-Ant-AndroidSdk-Haxe-zip GitHub et Driver ADB) :

  https://drive.google.com/open?id=1aVt0TaZ8mgc5z85QYneQXkDYa2FIMgUmAcgUtZQBIvM&authuser=0

- ## Développements :

  ### Préparation (à partir de la racine du zip téléchargé depuis github) :

  Aller dans ./haxeCordova/ et créer un "projet" flashdevelop ou autre IDE de votre choix.

  ### Structure:

  Le dossier de travail (casi) vide est dans **./haxeCordova/todo/**
  Les classes (casi) vierges sont dans **./haxeCordova/todo/template/src**
  Les autres templates sont dans **./haxeCordova/todo/template/**
  Les projets complets (norma et cordosample) déjà réalisés sont dans **./haxeCordova/done/**

  ### Commun aux 2 projets détaillés plus loin :

  Initialisation d'un projet :

- lancer start.bat sous Windows -ou autre shell avec les paths des softs accessibles.
- cd ./haxeCordova/todo/
- cordova create <dossier projet> <ext domaine>.<domaine>.<nom court app> "<nom appli>"
  ex:cordova create norma net.pixaline.norma "Norma Pedroche"
- cd <dossier projet>
- cordova platforms add android
- remplacer ./www/ par le www/ customisé de template/
- remplacer./platforms/android/res/drawable*/ par les drawable*/ customisés
- modifier www/index.html, ./config.xml et platforms/android/res/xml/config.xml selon besoins.

Commun (suite)

Test sur émulateur ou appareil mobile :

- lancer 1 des émulateurs de l'android SDK Manager
  (sous windows) ("C:/Program Files/Android/android-sdk/SDK Manager.exe")
- tools>manage AVDs>device Définitions>choisir>clone>>create AVD>
  modifier/bidouiller jusqu'à ce que ça démarre
- cd <dossier projet>
- cordova emulate android

OU MIEUX :

- Tester directement sur mobile (la partie ADB de l'installation doit être parfaitement effectuée):
- Fermer émulateur android -si besoin.
- désinstaller sur mobile la même appli installée depuis un autre ordi -si besoin.
- cd <dossier projet>
- cordova run android ou cordova build android pour juste tester la compil.
- pour installer sur mobile sans re-compiler :
  adb install ./platforms/android/ant-build/<appname>-debug-unaligned.apk

---

Projet I - ultra simple
Transformer n'importe quel responsive website en app mobile.

- Effectuer commun>Initialisation d'un projet
- modifier www/js/index.js en insérant dans *haxeAppStart: function() { }* :
  window.location.replace("<url du site web, responsive de préférence>");

- C'est tout ?...
  … Et oui, c'est tout ! Pas d'Haxe encore !

- Effectuer commun>Test sur mobile ou émulateur.

Projet II - Tests des “Plugins” Cordova

- Effectuer commun>Initialisation d’un projet
  avec nom d’app “Cordova sample” et dossier “cordosample/”

- Note:
  Sur dernière version cordova les apk ont comme nom “MainActivity” au lieu du nom de l’appli
  Pour changer cela,
    - modifier le classname de
      ..../platforms/android/src/net/pixaline/cordosample/**MainActivity.java**
      puis save as : <NomAppli>.java ; puis delete MainActivity.java
    - mettre le nom de l’appli dans android:name=”” de
      ..../platforms/android/AndroidManifest.xml

- créer cordosample/src/ -contiendra les sources Haxe
- créer build.hxml -compilation Haxe en js ; voir modèle dans templates/
  créer dans src/ Main.hx et SampleGeolocation.hx à partir de “template/” -en s’inspirant de done/cordosample/src si besoin.

- Note : Apperçu de l’architecture cordova/js et haxe:
  - cordova.js dans index.html n’existe pas dans www/js car dynamiquement chargé.
  - l’api “extern” de Haxe “cordovax” et une version light de mon api “apix” sont
    à partir de cordosample, dans :
    ../../classes/
    c-à-d depuis le root github dans **classes/**

- créer SampleCompass.hx sur le même principe ainsi que les images pour la boussole à prendre dans done/cordosample/www/img/

- Installer les “plugins” Geolocation et Compass sur pc:
  - cordova plugin add org.apache.cordova.geolocation
  - cordova plugin add org.apache.cordova.device-orientation

- Effectuer commun>Test sur mobile ou émulateur

- L’ajout des autres tests se fait sur le même principe :

- SampleCamera.hx avec
  - cordova plugin add org.apache.cordova.camera
- SampleDatabase.hx
- SampleEvent.hx (test des boutons et events spécifiques mobile)
  - Enlever le window.addEventListener de index.js
  - cordova plugin add org.apache.cordova.dialogs - alert,etc
  - cordova plugin add org.apache.cordova.network-information - on/offline
  - cordova plugin add org.apache.cordova.battery-status - battery
  - cordova plugin add org.apache.cordova.console - console.log()
    à utiliser éventuellement avec
    http://jsconsole.com/
    :listen
    copier/coller ligne donnée dans le <head> de index.html
- SampleDevice.hx avec
  - cordova plugin add org.apache.cordova.device
- SampleMotion.hx
  - cordova plugin add org.apache.cordova.device-motion
  - et ses images à prendre dans done/cordosample/www/img/
- SampleSOS.hx
  - cordova plugin add org.apache.cordova.vibration
- SampleContact.hx
  - cordova plugin add org.apache.cordova.contacts

- Projet “cerise sur gâteau” mais non moins intéressant :

  ## Audio et gestion de fichiers

  Notes

  - Sur certains mobiles, il faut débrancher le mobile après install (cordova run android), pour avoir correctement accès aux directories que ce soit les dossiers de l’application ou bien les dossiers “external” c-à-d les autres dossiers de la sd card.
    Les dossiers “external” de l’api File de cordova qui permettent de lire des fichiers externes à notre application, sont = null si câble branché.
    Les dossiers internes à notre appli comme File.dataDirectory ne fonctionnent sur android, que quand le câble est branché. Il faut donc utiliser File.externalDataDirectory pour écrire des fichiers persistants.

  - On ne peut pas utiliser la balise html5 <audio> avec par exemple
    audio.src=File.applicationDirectory + "www/sound/voix01.amr" ; audio.play() ;
    Par contre on peut faire un upload de File.applicationDirectory + "www/sound/voix01.amr"
    puis
    audio.src = "http://www.pixaline.net/intra/cordosample/sound/voix01.amr"; audio.play();
    On peut aussi utiliser le plugin Media de Cordova pour enregistrer un son puis le relire ; sans donc utiliser de balise <audio>. Et cela marche très bien.

  - Le dossier d’enregistrement de l’appli standard du mobile varie selon les machines.
    ex:
    "My%20Documents/My%20Recordings/" ; // sur HTC desire hd
    "Sounds/" ; // sur Samsung GT I8190

- SampleAudioLocal.hx
  - cordova plugin add org.apache.cordova.media
  - cordova plugin add org.apache.cordova.File
  - cordova plugin add org.apache.cordova.media-capture
- SampleAudioServer.hx
  - cordova plugin add org.apache.cordovafile-transfer
  - cordova plugin add org.apache.cordova.media
  - cordova plugin add org.apache.cordova.File

- Modèle dans done/ :
  - Le main du projet audio est dans src/audio/ et la compil haxe->js se fait par “build.audio.hxml”.
  - le main.js est le même. Autrement dit : Les 2 builds sont dans le même projet et on aura l’une ou l’autre appli installée sur son mobile mais pas en même temps.

Mise en ligne de l'application.

- Générer un APK "release"
  cordova build --release android
  ce qui produit un <NomAppli>-release-unsigned.apk dans .../platforms/android/ant-build

- Générer une clé privée pour l'appli :
  cd <dossier projet>/platforms/android/ant-build/

  keytool -genkey -v -keystore <my-release-key>.keystore -alias <alias_name> -keyalg RSA -keysize 2048 -validity 10000 (27 ans)
  Se souvenir de l'alias_name et du mot de passe choisi pour une prochaine version à mettre en ligne.
  Ex apixkey et azqswx2015 avec <my-release-key> == *cordosample.keystore*

- Signer l'apk :
  jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore <NomAppli>-release-unsigned.apk <alias_name>
  note : aucun fichier n'est créé
  pour + info : http://developer.android.com/tools/publishing/app-signing.html

- ZipAligner l'apk :
  Si besoin:
  copier .../Android/android-sdk/build-tools/<num version>/zipalign.exe
  dans …./Android/android-sdk/tools/zipalign.exe
  puis taper:
  zipalign -f -v 4 <NomAppli>-release-unsigned.apk <NomAppli>.apk

- Dernier test sur tél avant googlePlay
  adb install <NomAppli>.apk
  c'est fini pour la partie (aie) tech

Mise en ligne de l'application (suite)

- Préparation des images de promo, logo etc
  Voir .../todo/template/googleplay/ ou ../done/cordosample/googleplay/
  où se trouve les 4 formats du minimum syndical imposé.

- S'inscrire comme développeur Android "boutique"
  - Aller sur
    https://support.google.com/googleplay/android-developer/answer/6112435?hl=fr
    lire éventuellement la doc (en):
    https://developer.android.com/distribute/googleplay/developer-console.html

  - Créer ou utiliser un compte Google et se connecter.
    exemple avec cordosample@gmail.com & clavier+2015
  - Aller sur la création d'un compte "développeur"
    https://play.google.com/apps/publish/signup/
    ...Et là il faut préparer 25 US$ (c'est plus cher pour les dév. IOS)

- Puis suivre les instructions ; le processus est plutôt simple et bien fait.

--- Fin --- Bon courage ---