MODULE 5

Phonegap

OUTILS, INSTALLATION, CONFIGURATION

- **iOS**
  - Xcode et iOS SDK
    - Prérequis: Mac + Mac OS 10.9 Mavericks
    - Télécharger Xcode 5 depuis l'App Store
- Inclut tous les outils nécessaires (Xcode, simulateurs, SDK, ...)
  - Xcode 5 est nécessaire pour développer sous iOS7
  - Xcode 5 simplifie la gestion des clés, profils, et autres étapes fastidieuses liées au développement et au déploiement des applications iOS

ENREGISTREMENT "APPLE DEVELOPER"

- ***http://developer.apple.com*** > iOS Dev Center
- S'identifier ou s'inscrire (Create Apple ID)

INSCRIPTION À "L'APPLE DEVELOPER PROGRAM"

- Paiement annuel
- Non instantané (quelques jours)
- Rend possible le test des applications sur des appareil physiques (provisioning)
- Accès à un support technique
- Rend possible la distribution de ses applications, notamment sur l'App Store

CERTIFICATS ET PROVISIONING

- Developer certificate
- Distribution certificate
- Developer provisioning profile
- Distribution provisioning profile
- Bundle ID
- Processus manuel auparavant, désormais intégré à Xcode 5

OBTENTION DES CERTIFICATS SOUS XCODE 5

- Menu Xcode > Préférences > Accounts
- Ajouter un compte: "Add Apple ID" (cet identifiant doit être associé à un "Apple Developer Program")
- Dans la fenêtre du projet, onglet General, dans le champ "Team": choisir le compte précédemment créé
- Cliquer sur "Fix Issue" juste en dessous du champ précédemment renseigné
  - Créé et télécharge tous les profils et certificats nécessaires
    (visibles dans l'application "trousseau" (keychain))

TESTER SUR UN APPAREIL PHYSIQUE

- Provisioning profile (dev) nécessaire
- Ouvrir la fenêtre "Organizer" (menu Window > Organizer)
- Ouvrir l'onglet correspondant à l'appareil à utiliser
- Cliquer sur "Use for Development"
- Choisir un compte à utiliser pour le provisioning profile
- Ce profil sera automatiquement chargé après avoir construit et exécuté une fois l'application sur cet appareil
  (l'ajout manuel est également possible)

ANDROID (1)

- Télécharger le Bundle SDK ADT ***http://developer.android.com/sdk/index.html***
- Exécuter Eclipse
- Dans le SDK Manager (Window > Android SDK Manager)
  - Vérifier que les éléments suivants sont installés (installer les paquets manquants)
    - Android SDK Tools
    - Android SDK Platform-tools
    - Android SDK Build-tools
    - SDK Platform pour la ou les versions d'Android nécessaires
    - Les images système: ARM et Intel
  - Installer les paquets additionnels suivants (extras)
    - Google USB Driver (PC)
    - Intel x86 Emulator Accelerator (HAXM)
- Installer Intel HAXM (dans le dossier du SDK)

ANDROID (2)

- Créer un appareil virtuel
  - Ouvrir Android Virtual Device Manager (Window > Android Virtual Device Manager)
  - Dans l'onglet "Device Definitions", sélectionner "Nexus 4", puis "Create AVD"
  - Configurer l'AVD
  - Tester l'AVD créé: "Start", puis "Launch"
- Installer Google USB Driver (PC)
- Ajouter les chemins vers /platform-tools et /tools du SDK dans la variable d'environnement PATH.

NODEJS

- Télécharger sur ***nodejs.org***
- Installer avec les paramètres par défaut

PHONEGAP

- **Installer**
  - `npm install -g cordova`
    ou
    `sudo npm install -g cordova` (Mac OS)
- **Créer un projet**
  - `cordova create projet0 com.cnti-consulting.projet0 projet0`
    - nom du dossier
    - identifiant unique (communément nom de domaine inversé)
    - titre de l'application

AJOUT DES PLATEFORMES VOULUES

- `cordova platform add ios` (Mac)
- `cordova platform add android`
- ...
- Guide des plateformes:
  http://docs.phonegap.com/en/3.4.0/guide_platforms_index.md.html#Platform%20Guides

CONSTRUIRE (PREPARE + COMPILE)

- `cordova build`
- `cordova build ios`
- `cordova build android`

EXÉCUTER / TESTER L'APPLICATION

- iOS
  - via XCode:
    ouvrir le projet Xcode généré par le build (.xcodeproj)
  - Ligne de commande:

    installer ios-sim: `sudo npm install -ios-sim -g`

    lancer l'application dans le simulateur:
    `cordova emulate ios`
- Android
  - Sur un appareil connecté en USB:
    `cordova run android`
  - Dans l'émulateur android:
    `cordova emulate android`

TPS

- projet0
  créer une application de base Phonegap et l'exécuter dans un simulateur
- projet1
  convertir l'application du TP jQuery Mobile (Flickr) en application native

PLUGINS

- Ajouter un plugin
  - `cordova plugin add plugin1_id plugin2_id ...`

```
cordova plugin add
org.apache.cordova.device
org.apache.cordova.device-
motion
```

- Liste des plugins
  - *http://docs.phonegap.com/en/3.4.0/guide_cli_index.md.html#The%20Command-Line%20Interface_add_plugin_features*
- Informations sur l'appareil
- Connexion et batterie
- Accéléromètre
- Boussole
- Géolocalisation
- Camera
- Media
- Capture
- Fichiers (File System: appareil ou réseau)
- Notifications
- Contacts
- Globalisation
- Splashcreen
- Navigateur (inAppBrowser)
- Console

- Chercher un plugin
  - `cordova plugin search`

ÉVÈNEMENTS

- *http://docs.phonegap.com/en/3.4.0/cordova_events_events.md.html#Events*
- `deviceready`
- `pause`: l'application est placée en arrière plan
- `resume`: l'application est replacée au premier plan
- `backbutton`
  - Non supporté sur iOS
- `menubutton`
  - Non supporté sur iOS
- `searchbutton`
  - Android seulement
- ...

DEVICE PLUGIN

- *org.apache.cordova.device*
- *https://github.com/apache/cordova-plugin-device/blob/dev/doc/index.md*
- device.model
- device.platform
- device.uuid
- device.version

ACCÉLÉROMÈTRE (1)

- Capteur: détection des écarts de mouvement de l'appareil sur les 3 axes x, y et z
- *org.apache.cordova.device-motion*
- *https://github.com/apache/cordova-plugin-device-motion/blob/dev/doc/index.md*
- `navigator.accelerometer.getCurrentAcceleration(onSuccess, onError)`
  - L'objet acceleration est passé aux callbacks
    - `acceleration.x`
    - `acceleration.y`
    - `acceleration.z`
    - `acceleration.timestamp`
  - Non reconnu par iOS: cf. watchAcceleration

ACCÉLÉROMÈTRE (2)

- `var watchID = navigator.accelerometer.watchAcceleration(onSuccess, onError, options)`
  - L'objet acceleration est passé aux callbacks (idem getCurrentAcceleration)
  - options (optionnel):
    - `frequency`: intervalle en milli secondes. (défaut: 1000)
- `navigator.accelerometer.clearWatch(watchID)`

BOUSSOLE (1)

- Capteur: détection de l'orientation de l'appareil, évaluée de 0 (Nord) à 359.99.
- org.apache.cordova.device-orientation
- https://github.com/apache/cordova-plugin-device-orientation/blob/dev/doc/index.md
- navigator.compass.getCurrentHeading(onSuccess, onError)
  - L'objet heading est passé aux callbacks
    - heading.magneticHeading
    - heading.trueHeading
    - heading.headingAccuracy
    - heading.timestamp

BOUSSOLE (2)

- var watchID = navigator.compass.watchHeading(onSuccess, onError, options)
  - L'objet heading est passé aux callbacks (idem getCurrentHeading)
  - options (optionnel):
    - frequency: intervalle en milli secondes. (défaut: 100)
    - filter: seuil nécessaire pour déclencher l'évènement
- navigator.compass.clearWatch(watchID)

VIBREUR

- org.apache.cordova.vibration
- https://github.com/apache/cordova-plugin-vibration/blob/dev/doc/index.md
- navigator.notification.vibrate(time)
- time: durée de la vibration (ms)
- sur iOS, time est ignoré, et la durée est prédéfinie.

PHOTOS

- org.apache.cordova.camera
- https://github.com/apache/cordova-plugin-camera/blob/dev/doc/index.md
- navigator.camera.getPicture(onSuccess, onError, [options]);
- onSuccess(imageData)
  - imageURI : src = imageData
  - ou image encodée : src = "" + imageData;
- onError(message)

OPTIONS

- quality: 0-100
- destinationType: `Camera.DestinationType.DATA_URL (0) / .FILE_URI (1) / .NATIVE_URI (2)`
- sourceType: `Camera.PictureSourceType.PHOTOLIBRARY (0) / .CAMERA / .SAVEDPHOTOALBUM (2)`
- allowEdit: true/false
- encodingType: `Camera.EncodingType.JPEG (0) / .PNG (1)`
- targetWidth: (px)
- targetHeight: (px)
- mediaType: `Camera.MediaType.PICTURE (0) / .VIDEO (1) / .ALLMEDIA (2)`
- correctOrientation: true/false
- saveToPhotoAlbum: true/false
- popoverOptions: iPad seulement
- cameraDirection: `Camera.Direction.BACK (0) / .FRONT (1)`

CONFIG.XML

ICÔNES ET SPLASH SCREENS

SIGNATURE ET DISTRIBUTION SUR IOS

- Comprendre le processus
  - Pourquoi?
    - Certifier l'identité du développeur
    - Gérer des droits d'accès et la diffusion des applications
    - Limiter l'exécution d'applications (programme payant)

PROGRAMMES DÉVELOPPEUR

	iOS Developer Program	iOS Developer Enterprise Program
App Store		
In House		
Ad Hoc (100 appareils max)		
Tarif	$99 / an	$299 / an

COMPOSANTS

- Provisioning portal
- http://developer.apple.com/ios/manage/overview
- Certificats, UDID, Provisionings
  - iTunes Connect
- http://itunesconnect.apple.com
  - Back-office App Store

COMPOSANTS "TECHNIQUES"

Développeur	Apple
Clé privée	
Clé publique	Certificats
App ID	Provisionings
UDID	

CLÉ PRIVÉE, CLÉ PUBLIQUE

- Paire de clés RSA 2048 bits
- Assurer le décryptage par le possesseur de la clé privée uniquement
- Garantir l'identité de l'émetteur

CERTIFICATS (1)

- 2 types de certificats techniquement identiques
  (certificats certifiés par un tiers de confiance: Apple)
  - Développement: identifie un développeur
  - Distribution: identifie une équipe / un éditeur
- CSR: Certificate signing request
  - Généré à partir d'une clé privée
  - Anatomie:
    - Nom + Adresse email (chiffrés avec clé privée)
    - Clé publique (non chiffrée)
  - À partir de ce CSR, le provisioning portal peut générer un certificat:

CERTIFICATS (2)

- Contient les éléments du CSR (nom + mail + clé publique)
- est signé par Apple
- le tout est chiffré avec les clé privée d'Apple

APP ID

- Chaine de caractères identifiant une application (format reverse-DNS)
- 4526BLBB7Dcom.cnti-consulting.appName
  - Bundle Seed ID
    - Généré par Apple
    - Unique
    - Utilisable pour plusieurs applications
    - Utilisé pour accès keychain (login commun par exemple) et accessoires
  - Bundle ID
    - Généré par les développeurs
    - Libre et modifiable
    - Dans l'idéal, unique à chaque application

UDID

- Unique Device IDentifier: Identifiant unique de chaque appareil iOS
- 40 caractères (hexa)
- Ajout via l'iOS Provisioning Portal ou l'Organizer de Xcode
- Nombre d'UDID limité à 100 appareils pour le développement et la distribution Ad-Hoc

LE PROVISIONING

- Associe:
  - Une ou plusieurs identités (certificats)
  - Un App ID
  - Éventuellement des UDID
- "Cette application, signée par ces développeurs, peut être exécutée sur cet appareil"

RÉSUMÉ

- Clé privée + clé publique => Certificats
- Certificats + App ID (+ UDID) => Provisioning

MARCHE À SUIVRE (XCODE 5) (1)

- developer.apple.com/devcenter/ios/index.action
- Certificates, Identifiers & Profiles > Identifiers
  - Créer une nouvelle application
  - Choisir un compte associé à un certificat valide
  - Choisir un nom pour le Bundle ID (par convention, nom de domaine inversé: com.cnti-consulting.monApp-2014)
  - Choisir les services voulus (modifiable ultérieurement)
  - Confirmer et soumettre

MARCHE À SUIVRE (XCODE 5) (2)

- Certificates, Identifiers & Profiles > Provisioning Profiles
  - Ajouter un profil de distribution
  - choisir l'ID qui convient
  - puis le certificat correspondant
  - Nommer puis générer le profil
  - Le télécharger, double cliquer sur le fichier, Xcode s'ouvre et le profil est ajouté automatiquement

MARCHE À SUIVRE (XCODE 5) (3)

- iTunes Connect > Manage your apps
  - Ajouter une nouvelle application
  - langage
  - App Name: choisir un nom pour l'application
  - SKU Number: choisir un identifiant UNIQUE pour l'application
  - Bundle ID: choisir dans la liste
  - Date de disponibilité
  - Prix
  - Autres informations (version, copyright, catégories..)
  - Cliquer sur "View Details"
  - Cliquer sur "Ready to upload Binary"

MARCHE À SUIVRE (XCODE 5) (4)

- Dans Xcode
  - Dans la fenêtre du projet, onglet Info: mettre à jour l'identifiant du Bundle
  - onglet "Build settings" > Code signing identity: mettre à jour de "iOS developer" à "iOS distribution"
  - onglet Build settings > Provisioning Profile: choisir le profil correspondant (profil de distribution précédemment créé)
  - Nettoyer le projet (Menu Product > Clean)
  - Analyser le projet (Product > Analyze)

MARCHE À SUIVRE (XCODE 5) (5)

- Construire l'archive (Product > Archive)
- Dans la fenêtre "Organizer" qui apparaît, cliquer sur Validate et suivre les instructions
- Puis cliquer sur "Distribute" > "Submit to the iOS App Store", suivre les instructions, soumettre.
- iTunes Connect > Manage your apps
  - sélectionner l'application
  - vérifier le statut "Upload Received"

DISTRIBUTION ANDROID (1)

- Exporter l'application
  - Fichier APK
    - Archive ZIP contenant l'ensemble des dossiers et fichiers composant l'application
    - Peut être installée sur la plupart des appareils Android
- App manifest, Certificat, Code SDK nécessaire, Ressources, Application compilée...

DISTRIBUTION ANDROID (2)

- Processus
  - Créer un certificat
    - ATTN! ne pas utiliser le debug keystore par défaut, mais un keystore "release"
  - Créer l'archive "release"
    - cordova build android --release
- Créer un fichier keystore si aucun existant
  - keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -keyalg RSA -keysize 2048 -validity 10000

INSTALLER / DISTRIBUER L'APPLICATION

- Transférer manuellement le fichier APK sur les appareils
  - Attention sécurité
- Distribution commerciale
  - Google Play
  - Amazon
  - Nook
  - ...

GOOGLE PLAY STORE (1)

- play.google.com/store
- Accessible par environ 40% des appareils Android
- Google ne teste pas l'application! > publication rapide
- Vérifications et validation par les utilisateurs
- Développeur Google
  - https://play.google.com/apps/publish/signup/
- Soumission
  - Archive APK
  - Informations: Nom, Description, ...
  - Prix

GOOGLE PLAY STORE (2)

- Tableau de bord
  - Rapports
  - Nombre de téléchargement
  - Statistiques des téléchargements (OS, appareils, pays, ...)