DÉVELOPPEMENT DES APPLICATIONS MOBILES HYBRIDES MULTIPLATEFORMES

CH 2 - Installation de Ionic et première prise en main

FORMATEUR : MAHAMANE SALISSOU YAHAYA : ysalissou@gmail.com

(TechnoLAB -ISTA)

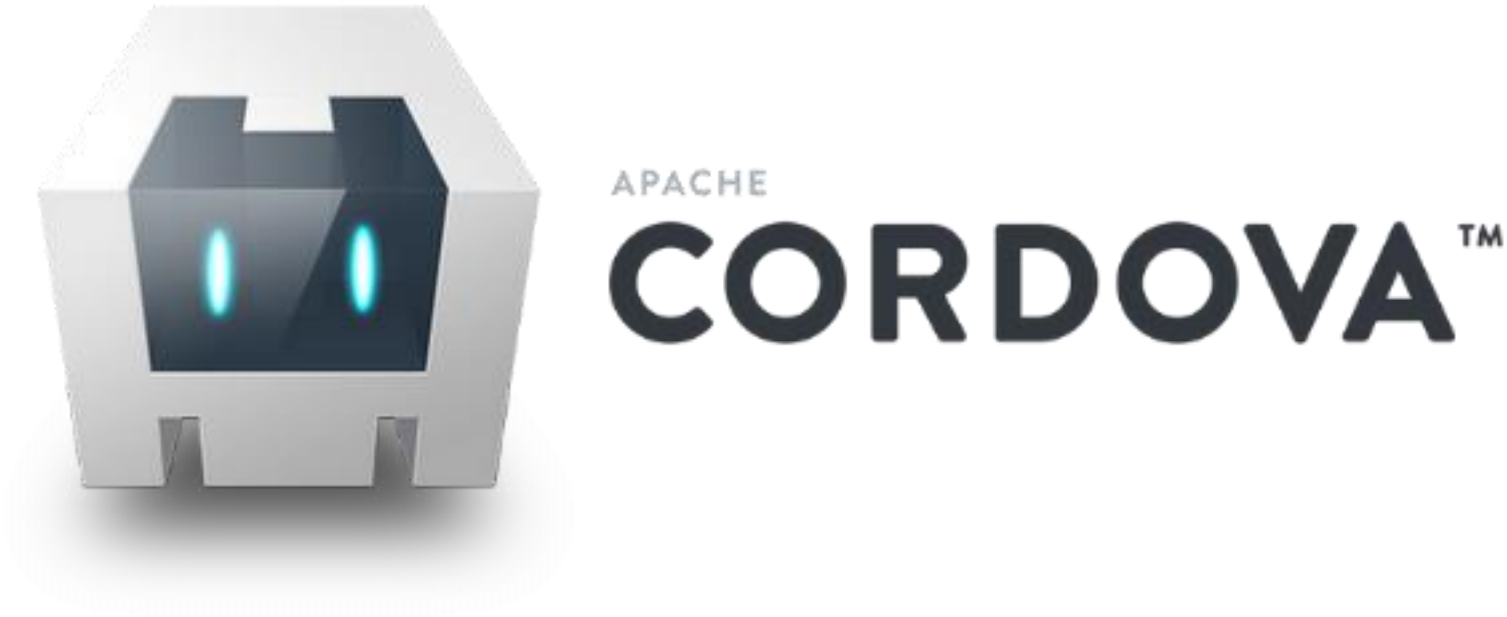
4ième année TEL

OCTOBRE 2021

- Ionic utilise un certain nombre d'outils permettant de créer rapidement une application mobile.
- Pris séparément, ils sont plus ou moins efficace, voir indépendant, mais mis en ensemble, ils sont d'une redoutable efficacité.
- Parmi ces outils nous pouvons citer principalement :

- **Ionic CLI** : c'est le couteau suisse de Ionic, un en de fonction disponible en ligne de commandes pour créer une application, la compiler, la déployer,...

- **Apache Cordova** : un framework open-source développé par la Fondation Apache. Il permet de créer des applications pour différentes plateformes en HTML, CSS et JavaScript.



- **NodeJS** : est un logiciel permettant de développer et d'exécuter du code JavaScript côté serveur, contrairement à ce qu'on a l'habitude de voir avec le javascript côté client.

- **NPM :** le gestionnaire de paquet de NodeJS

- **Angular :** un framework Javascript développé par Google

- **TypeScript** : un langage de programmation libre et open source développé par Microsoft qui a pour but d'améliorer et de sécuriser la production de code JavaScript.

- **SASS :** un langage de génération de feuilles de style (CSS dynamique)

NodeJS en bref

Depuis son commencement, JavaScript, a été, comme vous le savez très certainement, est un langage dit côté client.

Mais les choses ont quelque peu évolué avec NodeJS : cette technologie permet en effet d'executer du code écrit en JavaScript, aussi bien sur un navigateur (côté client), que côté serveur, tout comme des langages comme le Python ou encore le PHP.

De plus, NodeJS, à l'instar de Ionic, est Open Source, gratuit et disponible pour différentes plateformes (Windows, Linux, Unix, Mac OS,...)

NPM : Node Package Manager

Comme son nom peut le suggérer, **NPM** est le gestionnaire de paquet de NodeJS, qui étant très modulaire, voit son écosystème constamment enrichi par des modules développés par les membres de sa large communauté.

Installation

Windows et Mac OS

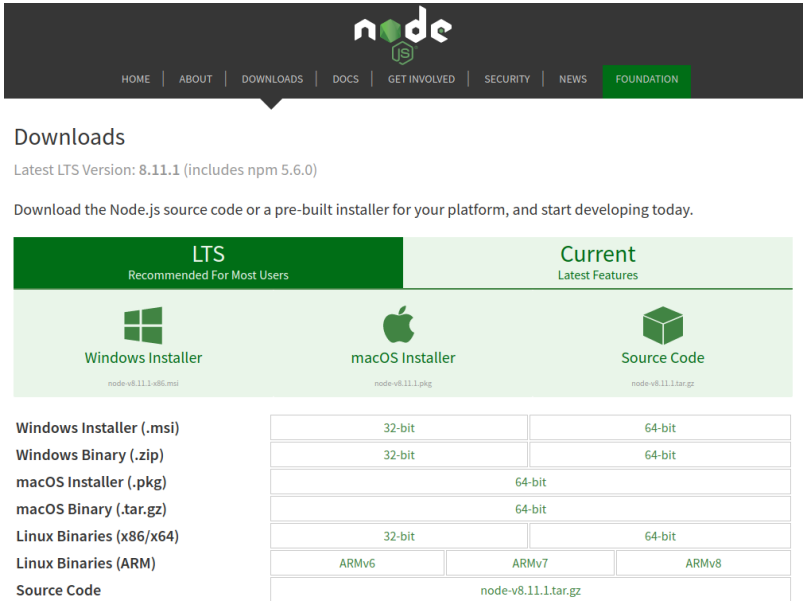
Pour installer NodeJS, il suffit simplement d'aller à l'adresse : **https://nodejs.org/en/download/**, de télécharger le gestionnaire d'installation au format .**msi** pour windows et .**pkg** pour Mac OS.

Laissez-vous ensuite simplement guider. Le gestionnaire installera également NPM.

Ouvrez un invite de commande et saisissez :

node -v

Vous devriez voir s'afficher la version actuelle de NodeJs.



Linux (Ubuntu)

sous linux et en particulier Ubuntu, il vous suffit de saisir les commandes suivantes depuis un invite de commandes :

```
$ sudo apt-get update
$ sudo apt-get install nodejs npm
```

Une fois Node et NPM installés, le reste se passera en ligne de commandes. Ouvrez donc votre terminal préféré et saisissez les commandes suivantes pour installer Ionic et Cordova :

$ sudo npm install -g ionic cordova

Le paramètre "**-g**" permet une installation globale de ces outils. De cette manière, vous n'aurez pas besoin d'être dans un répertoire particulier pour utiliser les commandes ionic ou cordova, sauf pour des actions comme la compilation qui requiert d'être à l'intérieur d'un projet.

Ce paramètre implique aussi que vous devrez lancer les commandes précédentes en tant qu'Admin sous Windows (clic-droit, démarrer l'invite de commande en tant qu'administrateur) et que sous Linux, vous êtes obligé d'utiliser le "**sudo**".

Avant d'aller plus loin, il sera nécessaire d'installer d'autres logiciels comme le SDK de Java ou celui d'Android.

Java SDK

Windows et Mac OS
Pour installer le SDK de Java sous Windows et Mac, il vous suffit de visiter le site de l'entreprise Oracle, qui détient et maintient le logiciel : ***http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html*** et de choisir le fichier (.exe pour Windows ou .dmg pour MacOS) adapté à votre machine (32 ou 64 bits)

Linux

Mise à jour des dépôts
$ sudo add-apt-repository ppa:openjdk-r/ppa
$ sudo apt-get update

Installation d'OpenJDK

sudo apt-get install openjdk-8-jdk

Android SDK

La meilleur façon d'installer le SDK d'Android est encore d'installer Android Studio. Pour ce faire, rien de plus simple, il suffit de visiter le site ***https://developer.android.com/studio/index.html#downloads*** et de télécharger le paquet associé à votre OS.

Windows

Une fois le téléchargement effectué, vous n'aurez plus qu'à lancer l'installation en cliquant sur le fichier au format .exe et suivre le setup. L'installation du SDK se fera en même temps.

Voilà, c'est tout.

Mac OS

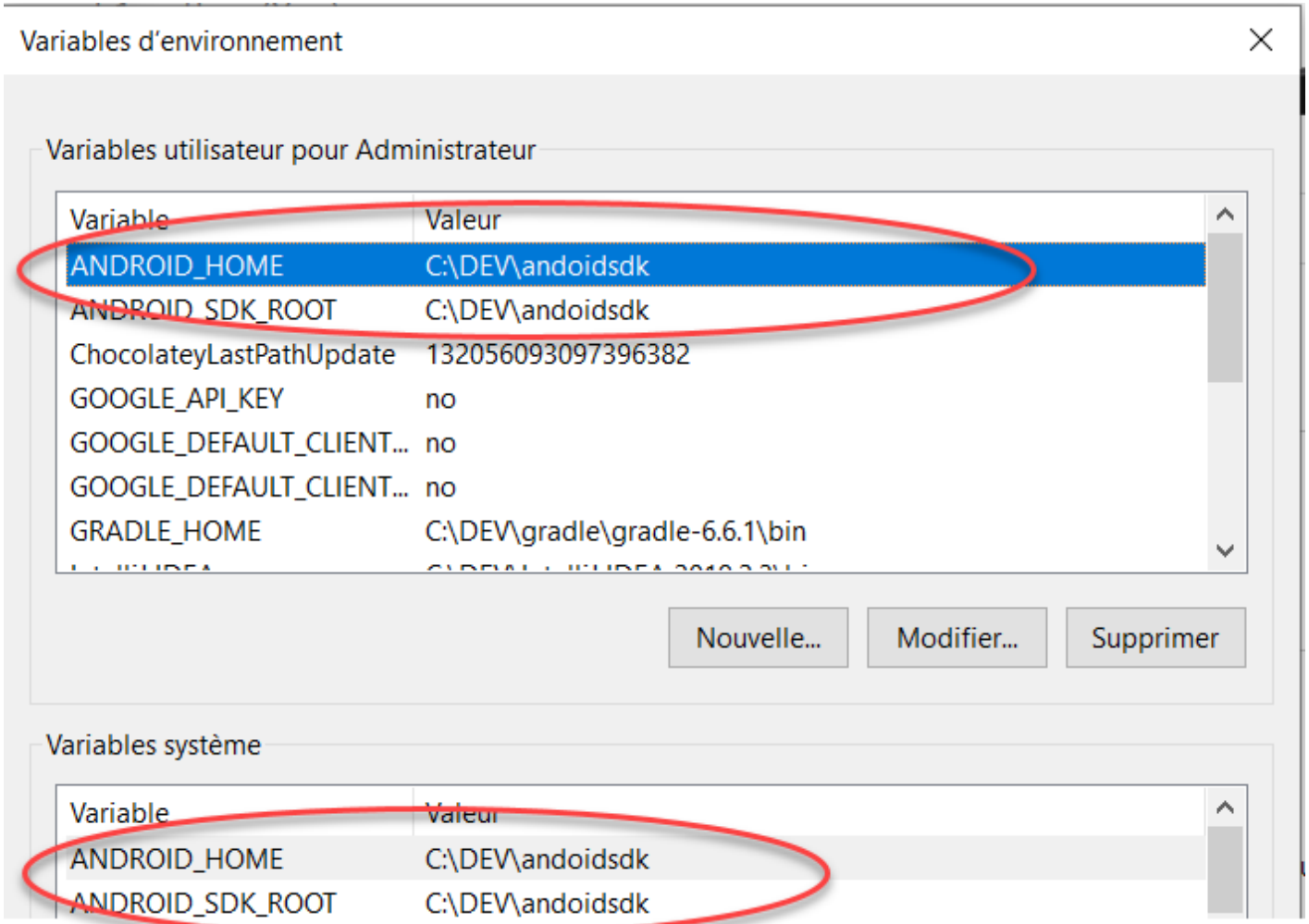
Lancer l'installation en cliquant sur le fichier au format .dmg téléchargé précédemment.
Glisser-déposer (Drag-n-drop) ensuite Android Studio dans le dossier Applications
Le setup devrait ensuite finaliser l'installation du SDK
Voilà.

Linux

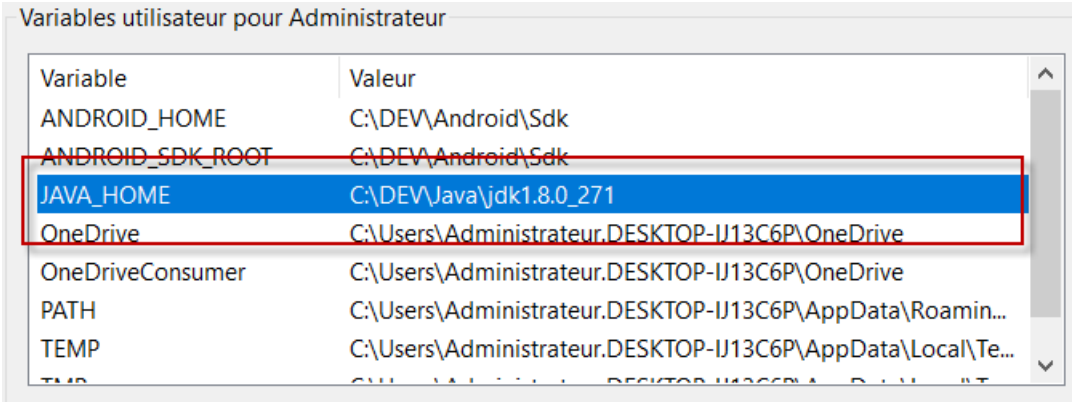
Décompresser le fichier .zip téléchargé précédemment dans un dossier approprié. Je vous propose le dossier **/opt/** de manière à le partager entre les différents acteurs de votre OS.
Ouvrez un invite de commandes (CTRL + ALT + T) et exécuter le fichier **/opt/android-studio/bin/studio.sh**.
Suivez le setup
Si votre OS est une machine 64-bit, vous aller devoir installer quelques dépendances logicielles :

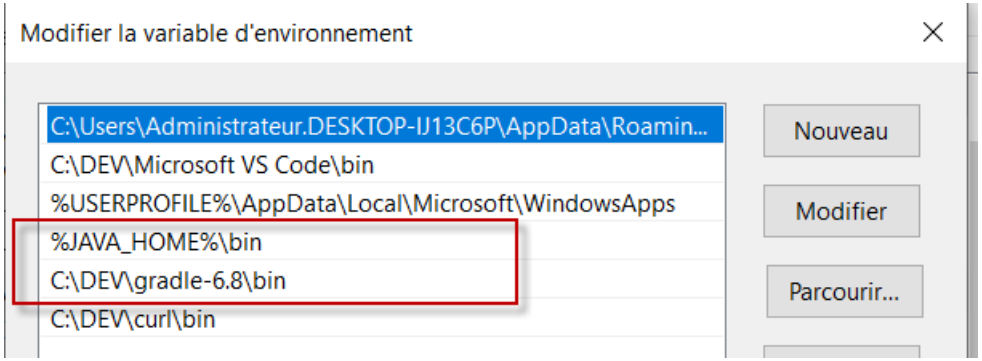
```
$ sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.0:i386
```

Android SDK – Variable d'environnement



JAVA_HOME & GRADLE





Git

Ionic utilise le gestionnaire de dépôt Git dans son workflow de développement actuel. Pour l'installer, rien de plus simple, il vous suffit d'aller à la page de téléchargement suivante : ***https://git-scm.com/downloads*** et choisir le paquet correspondant à votre OS.

Sous Linux, il est également possible de l'installer en saisissant simplement la commande :

$ apt install git

Éditeur de code

Vous pouvez bien évidemment utiliser l'éditeur de votre choix, mais s'il vous vient l'envie de tester autre chose, je vous propose ici deux éditeurs de code intéressants pour développer avec Ionic.

Visual Studio Code

Éditeur de code extensible développé par Microsoft pour Windows, Linux et OS X. C'est l'éditeur que je recommande le mieux, car il vous facilitera énormément la vie, grâce notamment à des composants proposés par la communauté des développeurs Ionic à installer directement dans l'éditeur.

Télécharger : ***https://code.visualstudio.com/***

Atom
Éditeur de texte libre pour OS X, GNU/Linux et Windows développé par GitHub.

Télécharger : ***https://atom.io/***

Cette étape va nous permettre de disposer d'un compte sur le cloud de Ionic. On en parle en détails au Chapitre 11, mais globalement, Ionic Cloud permet de :

- Compiler une application sans devoir installer sur son local toutes les dépendances nécessaires à cette compilation.
- Faire tester votre application en avant-première via l'application Ionic View1
- Gérer les erreurs levées dans l'application mobile
- ...

La création d'un compte est gratuite et pour ce faire, rendez-vous à l'adresse ***https://dashboard.ionicjs.com/signup*** et remplissez le formulaire pour compléter votre inscription.

Connectez-vous ensuite à Ionic PRO et cliquez sur le bouton "New app" pour créer une application que l'on liera plus tard à notre application mobile.

Donnez un nom à cette nouvelle app.

Une fois l'application créée, un identifiant unique lui est attribuée. Cet identifiant nous sera utile à la création de notre application depuis notre poste de travail.

Voilà, vous y êtes. On va donc pouvoir créer notre première application mobile.

Cette étape va nous permettre de disposer d'un compte sur le cloud de Ionic. On en parle en détails au Chapitre 11, mais globalement, Ionic Cloud permet de :

- Compiler une application sans devoir installer sur son local toutes les dépendances nécessaires à cette compilation.
- Faire tester votre application en avant-première via l'application Ionic View1
- Gérer les erreurs levées dans l'application mobile
- ...

La création d'un compte est gratuite et pour ce faire, rendez-vous à l'adresse ***https://dashboard.ionicjs.com/signup*** et remplissez le formulaire pour compléter votre inscription.

Connectez-vous ensuite à Ionic PRO et cliquez sur le bouton "New app" pour créer une application que l'on liera plus tard à notre application mobile.

Donnez un nom à cette nouvelle app.

Une fois l'application créée, un identifiant unique lui est attribuée. Cet identifiant nous sera utile à la création de notre application depuis notre poste de travail.

Voilà, vous y êtes. On va donc pouvoir créer notre première application mobile.

Pour créer votre première application, rien de plus simple, il suffit de saisir la commande suivante depuis votre invite de commandes :

$ ionic start monAppli blank
✔ Creating directory ./monAppli - done!
✔ Downloading and extracting tabs starter - done!

? Would you like to integrate your new app with Cordova to target native iOS and Android? (y/N)
A la question "Would you like to integrate your new app with Cordova to target native iOS and Android?" saisir "y".

Et à la question "Install the free Ionic Pro SDK and connect your app?" , répondez aussi par un "y".

Vous allez devoir entrez vos identifiants et générer une paire clé privé/publique en choisissant "Automatically setup new a SSH key pair for Ionic Pro«

Suivez ensuite le setup et garder les valeurs par défaut (choisir "Y" à chaque fois).

La commande va créer une application la plus "vide" possible (blank). Mais il également possible de créer une application avec des onglet, un menu latéral et bien d'autres encore (voir la commande ionic start --list décrite dans la suite)

La syntaxe générique de création d'une application est la suivante :

$ ionic start [<name>] [<template>]

Entrée	Description
name	C'est le nom de votre application au format Camel par ex. Vous pouvez également l'écrire tout en minuscule (ce que je recommande)
template	C'est le template ionic de votre choix. Pour afficher la liste des templates disponible actuellement, vous pouvez saisir la commande **ionic start --list** (voir ci-dessous)

$ ionic start --list
tabs ..... ionic-angular A starting project with a simple tabbed interface
blank ..... ionic-angular A blank starter project
sidemenu ..... ionic-angular A starting project with a side menu with navigation in the content area
super ..... ionic-angular A starting project complete with pre-built pages, providers and best practices for Ionic development.
conference ..... ionic-angular A project that demonstrates a realworld application
tutorial ..... ionic-angular A tutorial based project that goes along with the Ionic documentation
aws ..... ionic-angular AWS Mobile Hub Starter
sidemenu ..... ionic1 A starting project for Ionic using a side menu with navigation in the content area
maps ..... ionic1 An Ionic starter project using Google Maps and a side menu

Il est également possible de créer une application à partir d'un dépot git :

$ ionic start monappli_sur_git https://github.com/charlesen/monappli_sur_git

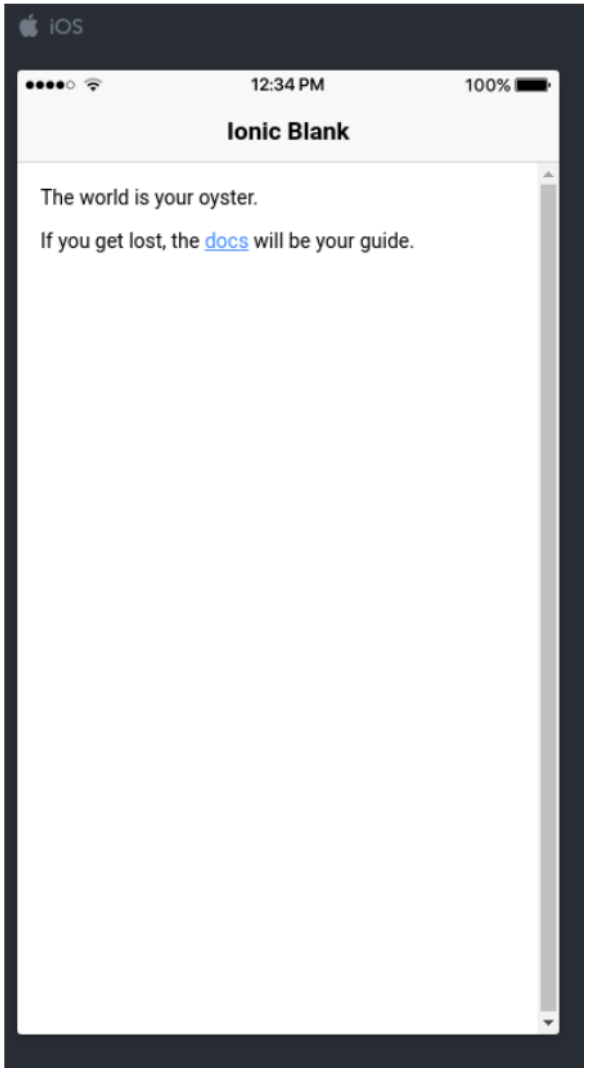
Une fois votre application créée, accédez au dossier nouvellement créé, puis démarrer le projet :

$ cd monappli
$ ionic lab
S'il vous ait demandé d'installer Ionic PRO, choisissez "Y".

Ionic devrait ensuite ouvrir votre application depuis votre navigateur préféré.

Félicitations, vous avez créé votre première application mobile !

Dans la suite du livre nous allons progressivement aborder des notions plus complexe du Framework en partant d'un exemple concret : la création d'une application mobile pour la gestion d'un trombinoscope du établissement nommé **TrombiSchool**.



Création du projet

Comme nous l'avons vu, ouvrez donc votre terminal et saisissez les commandes suivantes :

ionic start trombischool

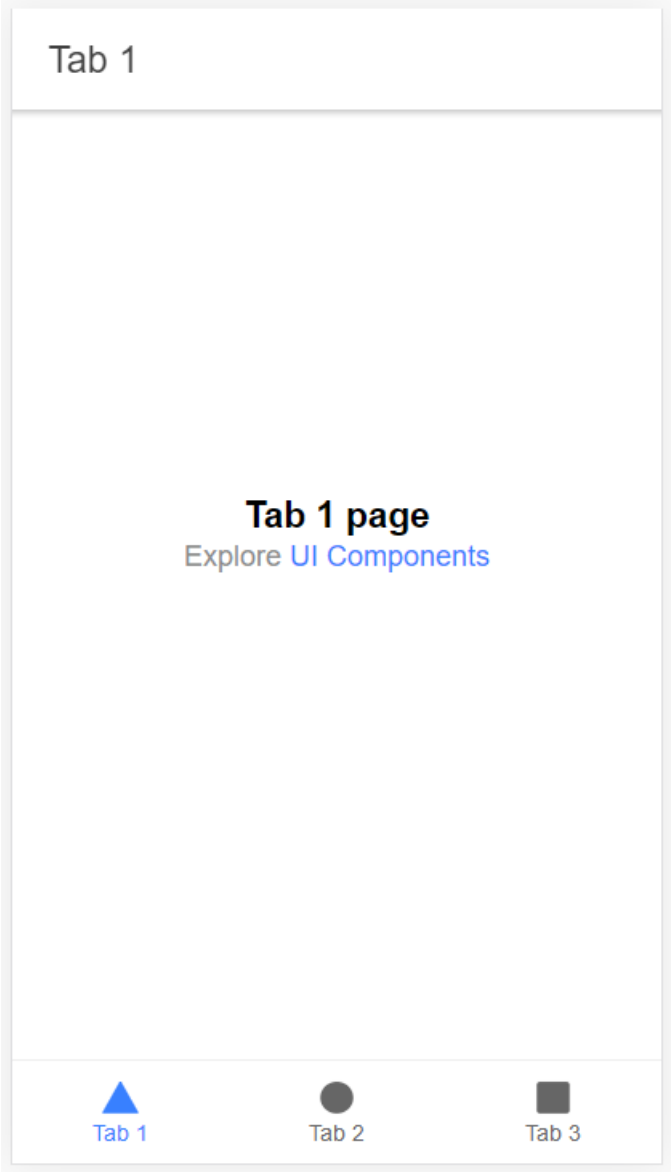
Démarrez ensuite l'application avec la commande serve :

$ cd trombischool
$ ionic serve
$ ionic –lab

Ajoutons ensuite les plateformes Android et iOS à notre projet comme ceci :

$ ionic cordova platform add android
$ ionic cordova platform add ios

$ ionic cordova run android



Faisons un peu le tour de l'anatomie d'un projet type sous Ionic. :

config.xml

Ce fichier est utilisé pour gérer toute la configuration de la partie native de l'application. C'est ici par exemple que vous rajouterez les autorisations nécessaires pour l'utilisation de la Camera dans votre application. Chaque plugin ajouté à votre application sera rajouté automatiquement à ce fichier.

C'est aussi dans ce fichier que vous devriez renseigner le numéro de version de votre application, utile dans l'étape de publication sur les stores (Google ou Apple Store par exemple)

```
<?xml version='1.0' encoding='utf-8'?>
<widget id="io.ionic.starter" version="0.0.1" xmlns="http://www.w3.org/ns/widgets" xmlns:cdv="http
    <name>monAppli</name>
    <description>Une application de test</description>
    <author email="hello@charlesen.fr" href="http://ionicframework.com/">Charles EDOU NZE</author>
    <content src="index.html" />
    <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:*" />
    <preference name="ScrollEnabled" value="false" />
    <preference name="android-minSdkVersion" value="19" />
    ...
```

ionic.config.json

Ce fichier contient des informations de base sur votre application (nom de l'application, App ID...), et est utilisé notamment pour uploader votre application sur le cloud Ionic.

package.json

C'est le fichier de configuration de Node. A la création d'un projet, Ionic lance automatiquement la commande **npm install**, qui va installer un certain nombre de paquets en arrière-plan et stockés dans le **dossier node_modules**. C'est grâce à ce fichier package.json que npm sait quel paquet installé dans le projet

./platforms

Ce dossier contient les versions "**natives**" de l'application. Un dossier a été ajouté pour chaque plateforme cible. Pour afficher la liste des plateformes actuellement supportées, il suffit de faire :

$ ionic cordova platform list

```
> cordova platform ls
Installed platforms:
  android 7.0.0
  ios 4.5.4
Available platforms:
  browser ~5.0.1
  ios ~4.5.4
  osx ~4.0.1
  windows ~5.0.0
  www ^3.12.0
```

./plugins
Le dossier contient tous les plugins Cordova utilisés par l'application. Pour rappel, un plugin un conteneur faisant appel à des fonctions natives (voir repertoire platforms/[NOM_PLATEFORME]/CordovaLib) à partir d'un code JavaScript.

./resources
Ce dossier contient les différentes icônes de l'application et le splashscreen (image au chargement de l'application). La commande ionic cordova resources permet la génération des icônes pour différents types de format d'écran.

$ **ionic cordova resources**
✔ Collecting resource configuration and source images - done!
✔ Filtering out image resources that do not need regeneration - done!
✔ Uploading source images to prepare for transformations - done!
✔ Generating platform resources: 50 / 50 complete - done!
✔ Modifying config.xml to add new image resources - done!

./src/
C'est à l'intérieur que l'on retrouve le code de l'application à proprement. Lorsque l'on voudra rajouter de nouveaux écrans, de la logique métier,...c'est ici que cela se passera.

On retrouve du code écrit en TypeScript (nous en reparlerons en détails au chapitre 6) dont l'extension de fichiers est .ts.

On retrouve aussi du html, du css,...

Écran Mobile = 1 Fichier .ts + 1 Fichier .html + 1 Fichier .scss

le fichier **src/app/app.module.ts** est le point d'entrée métier de notre application.

```
@NgModule({
  declarations: [MyApp, ContactPage, HomePage],
  imports: [BrowserModule, IonicModule.forRoot(MyApp)],
  bootstrap: [IonicApp],
  entryComponents: [MyApp, ContactPage, HomePage],
  providers: []
})
export class AppModule {}
```

C'est dans ce fichier que l'on décide quelle composant (ici MyApp) sera le composant principal. On expliquera ces notions de composants dans le chapitre 8.

Dans le chapitre suivant, nous allons apprendre à customiser notre application pour qu'elle soit un peu plus à notre image. Mais en attendant, exerçons-vous un peu.

./src/index.html
C'est l'entrée principale du projet. Il faut se rappeler qu'une application Hybride utilise la technologie WebView du téléphone qui se comporte alors comme un mini-navigateur à l'intérieur duquel on peut afficher un site web, qui est votre projet.

A l'intérieur de ce fichier, Ionic va aller chercher le tag <ion-app> à l'intérieur duquel vos différents écrans seront chargés.

```
<ion-app></ion-app>
```

tsconfig.json et tslint.json
Ces fichiers sont utilisé par TypeScript et décrivent notamment la manière dont celui-ci doit être compilé. Vous n'aurez pas nécessairement de les configurer vous-même, les valeurs par défaut étant suffisantes.

./www
Ce dossier, qui est auto-généré, contient la version web de votre application mobile. C'est grâce à ce dossier que vous pouvez visualiser l'application depuis un navigateur.