React Native  
Watch Next

# Démarche d’expérimentation

## Ionic[[1]](#endnote-1)

### Description

Ionic est un ensemble d’outils qui permet de développer des applications mobiles hybrides rapidement et facilement avec AngularJS[[2]](#endnote-2) et Cordova[[3]](#endnote-3). Cet ensemble permet également un maintien constant de l’application comparativement à une application native conventionnelle. Elle s’appuie sur la plateforme NodeJS[[4]](#endnote-4) et NPM[[5]](#endnote-5) pour installer les modules utilisés.

### Problèmes rencontrés

Le problème avec Ionic dans le cadre de notre projet est le manque de composants ainsi que les contraintes imposées par la plateforme web. Nous voulions faire un lecteur vidéo[[6]](#endnote-6) personnalisé pour accomplir une fonctionnalité spécifique, mais nous étions limités au lecteur par défaut de Ionic qui n’était vraiment pas convenable pour notre application. Nous devions donc trouver un nouveau Framework nous offrant plus de possibilités de personnalisation ainsi qu’une meilleure gestion de la performance.

## React Native[[7]](#endnote-7)

### Description

React Native est un Framework mobile qui a été développé par Facebook. Il offre un très grand support en ligne par sa popularité et les contributeurs sur GitHub ne cessent de croitre. Il offre une approche par composants très intéressante. Le point fort de React Native est qu’il est basé sur un état changeable de DOM virtuel qui facilite l’ajout et la modification d’éléments au fur et à mesure du développement. En d’autres termes, il n’est pas nécessaire de constamment recompiler l’application puisque les nouveaux éléments apparaissent instantanément sur le périphérique. De plus, le DOM virtuel offre un outil d’inspection semblable à l’inspecteur sur une page web qui nous a grandement facilité la tâche lorsqu’est venu le temps de personnaliser l’affichage.

### Résolution du problème initial

React Native nous a permis de remédier aux deux problèmes majeurs rencontrés lors de notre première expérimentation avec Ionic. L’approche par composants nous a permis d’utiliser facilement une librairie open source qui s’appelle React native-youtube[[8]](#endnote-8) . Nous avons pu personnaliser le lecteur vidéo à notre guise pour ensuite l’intégrer facilement à notre application. Par la suite, nous avons cherché à appliquer un effet de « swipe » (balayer) sur la vidéo qui permet d’interagir avec celle-ci. La librairie React-native-swipeout[[9]](#endnote-9) était la solution à notre problème, mais elle n’a pas été évidente à mettre en place dans notre projet. Il a fallu modifier le code initial de la librairie pour atteindre le résultat désiré.

Nous pensons arriver à proposer une application pleinement fonctionnelle d’ici la fin de la session et les résultats jusqu’à maintenant sont très satisfaisants. Les outils d’expérimentation développés grâce aux librairies précédemment mentionnées nous ont permis de résoudre les incertitudes apportées par Ionic.

Certains éléments comme la base de données n’ont pas encore été exploités à leurs pleins potentiels, mais nous allons travailler très fort pour y arriver dans les prochaines semaines. Nos expérimentations ressemblent à ce que nous avions imaginé, mais il nous reste encore plusieurs éléments à tester du côté de React Native et la courbe d’apprentissage est bien plus longue qu’avec Ionic. Nous croyons cependant que le tout en vaut la peine puisqu’une fois pris en main, React Native se révèle comme étant un outil incroyablement puissant qui apporte la vitesse d’une application native tout en ayant la flexibilité d’une application hybride.

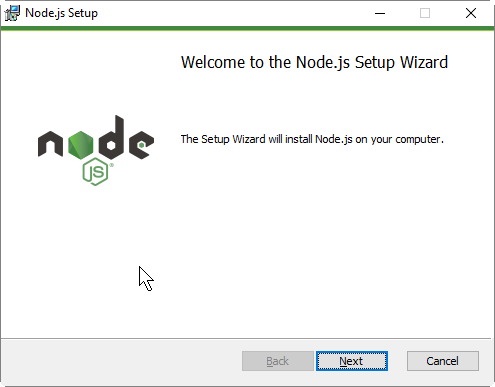
« Learn once, write anywhere » (Slogan de React Native)

# Devis technique de l’outil

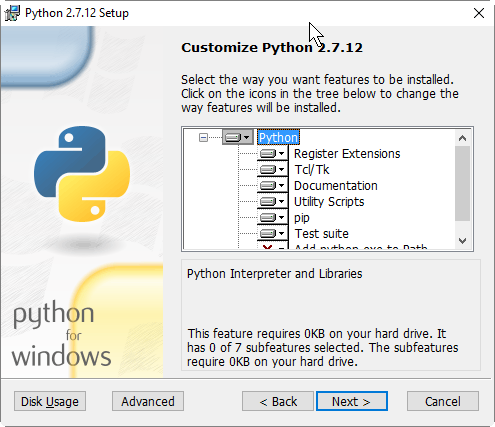
## Préinstallation des composantes

Windows (Voir plus bas pour procédure sur Mac OS)

##### Installation de Node.JS

Installation par un exécutable avec ce lien : https://nodejs.org/en/download/ 

##### Installation de Python2

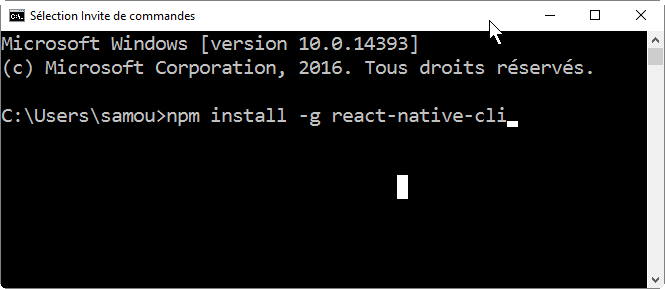
Installation par un exécutable avec ce lien : [Python\_v2712](https://www.python.org/ftp/python/2.7.12/python-2.7.12.msi)

##### Installation de REACT NATIVE

###### Installation avec Node.js

Ouvrir Node.js et taper cette commande (npm est préinstallé dans Node.js)

npm install -g react-native-cli



#### Installation des outils connexes à l’utilisation de REACT NATIVE

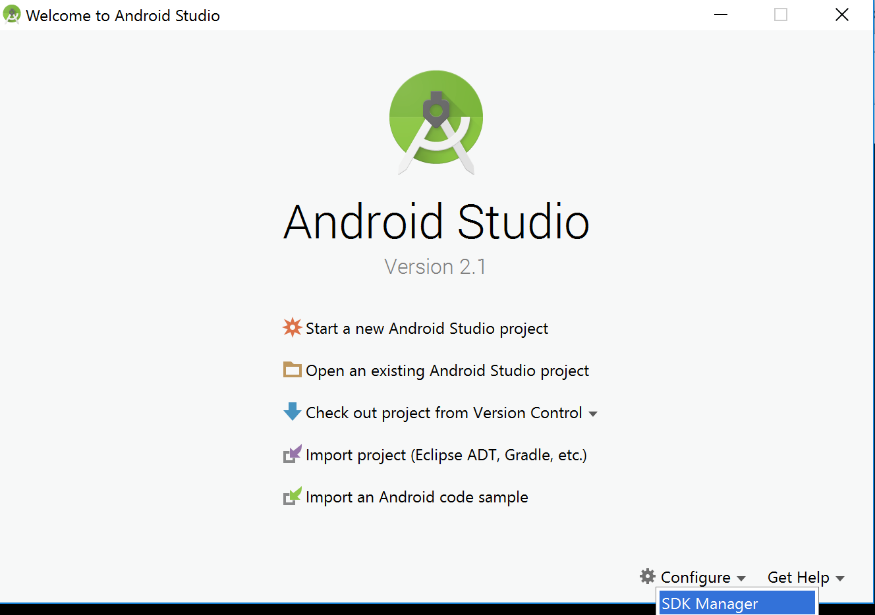
##### Installer Android Studio

###### Installation avec l’exécutable d’Android Studio : [Android\_Studio\_2220](https://developer.android.com/studio/index.html#win-bundle)

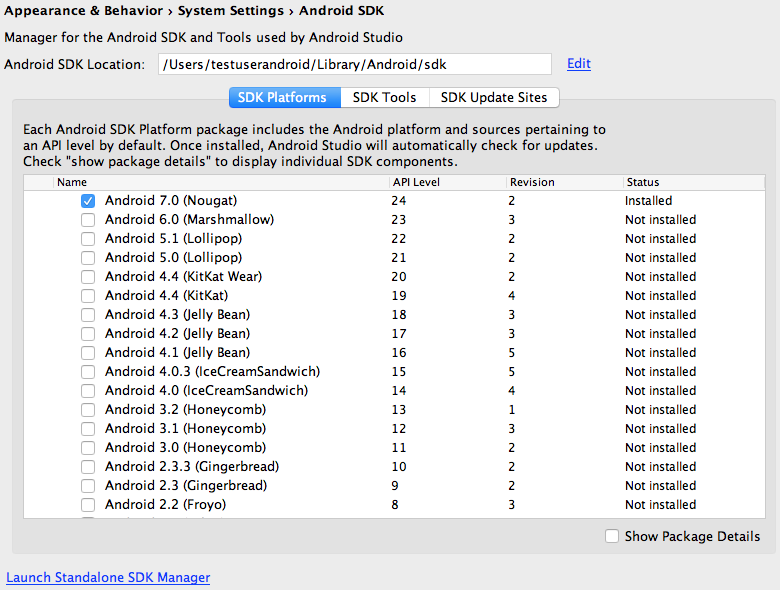
Vidéo explicative [ici](https://developer.android.com/studio/install.html)

###### Vérifier si Android SDK a été bien installé

Ouvrir Android studio et aller dans l’option Configure –SDK Manager

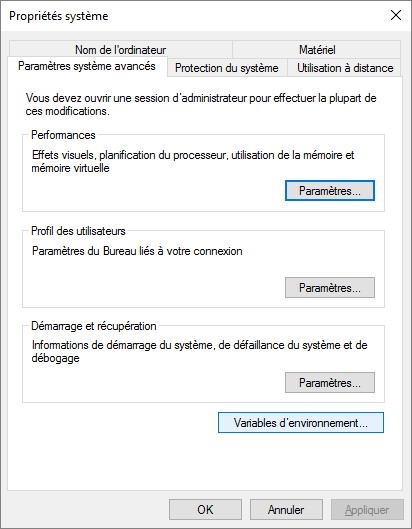
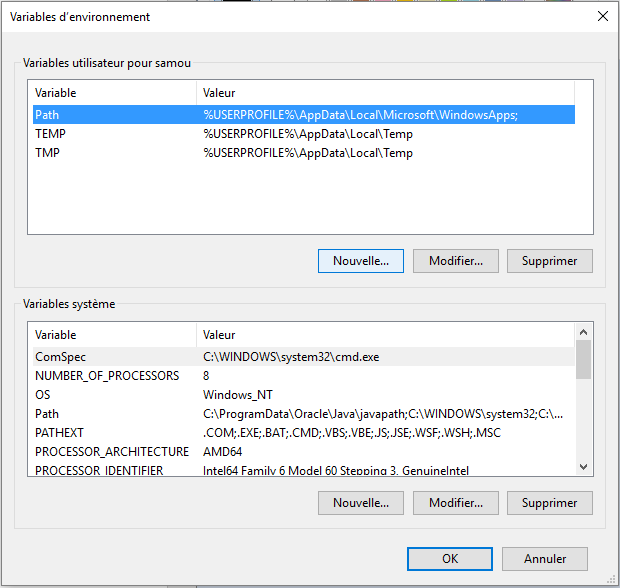


Vérifier s’il y a bien un SDK d’installé  
(Il est préférable d’installé la version récente et la version courante ex : Nougat 7 et Marshmallow 6)



###### Configurer les variables d’environnement

Écrire dans la barre de recherche Windows : Modifier les variables d’environnement système ou sysdm.cpl

Cliquer sur « Nouvelle … » dans le variables système.

Nom de la variable : ANDROID\_HOME

Valeur de la variable : C:\Users\[votre nom d’utilisateur]\AppData\Local\Android\sdk

###### Configurer votre machine Android

Ouvrir cmd.exe et marquer « android avd »

S’il n’y en a pas, créez-vous un Android Virtual Device (AVD) en suivant ce guide : [Create and Manage Virtual Devices](https://developer.android.com/studio/run/managing-avds.html)

Ou

Utiliser votre téléphone en le connectant à votre ordinateur à l’aide du câble USB.[[10]](#footnote-1)

Mac OS (Beaucoup plus facile qu’Android)

Installer [Homebrew](http://brew.sh/) :

/usr/bin/ruby -e "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install>)"

Installer Node et Watchman :

brew install node

brew install watchman

Installer React Native CLI :

npm install -g react-native-cli

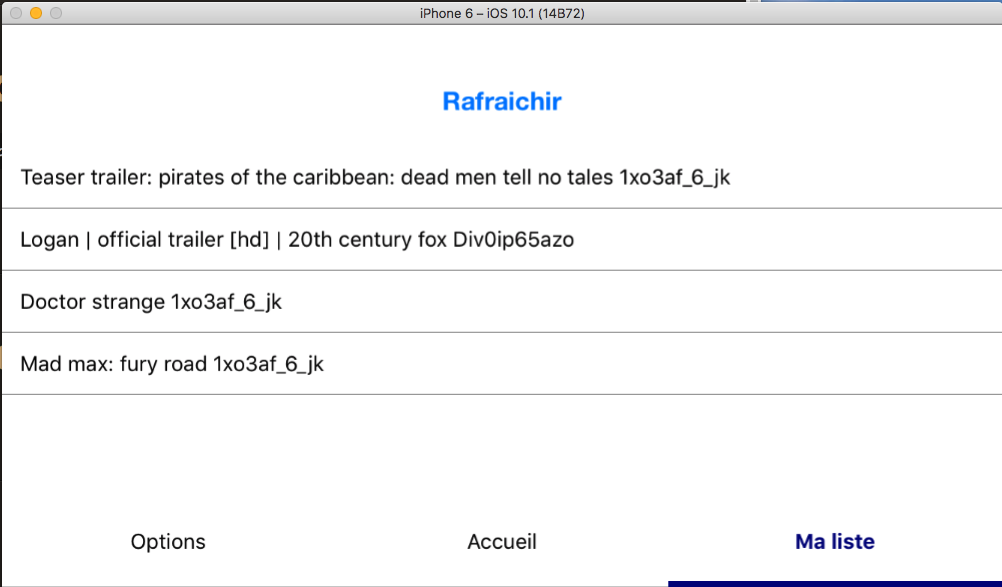
Installer la dernière version de xCode à partir de l’App Store :

<https://itunes.apple.com/us/app/xcode/id497799835?mt=12>

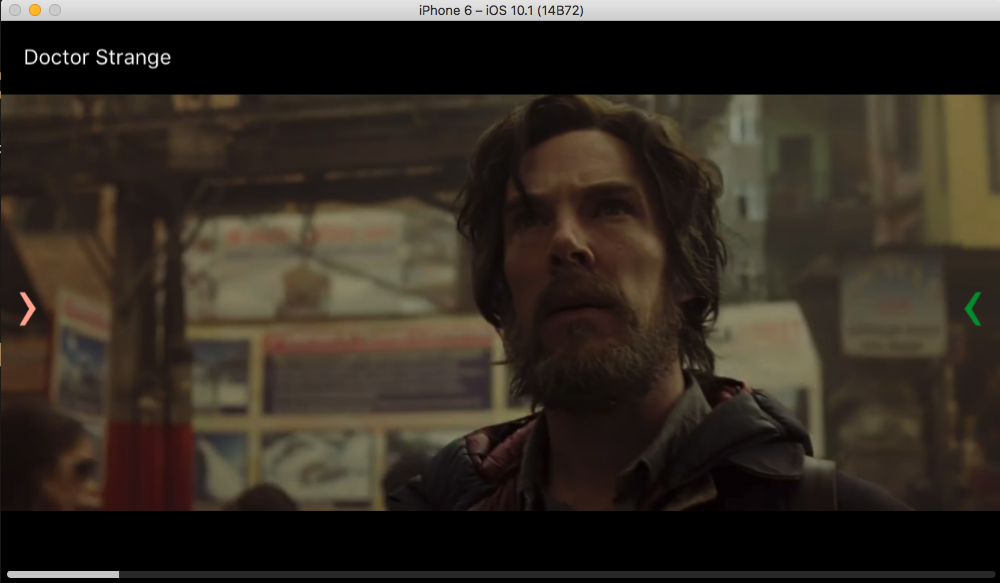
**Terminé!**

On peut maintenant lancer le projet à partir de « WatchNext.xcodeproj » qui se trouve dans le dossier « ios » du projet.

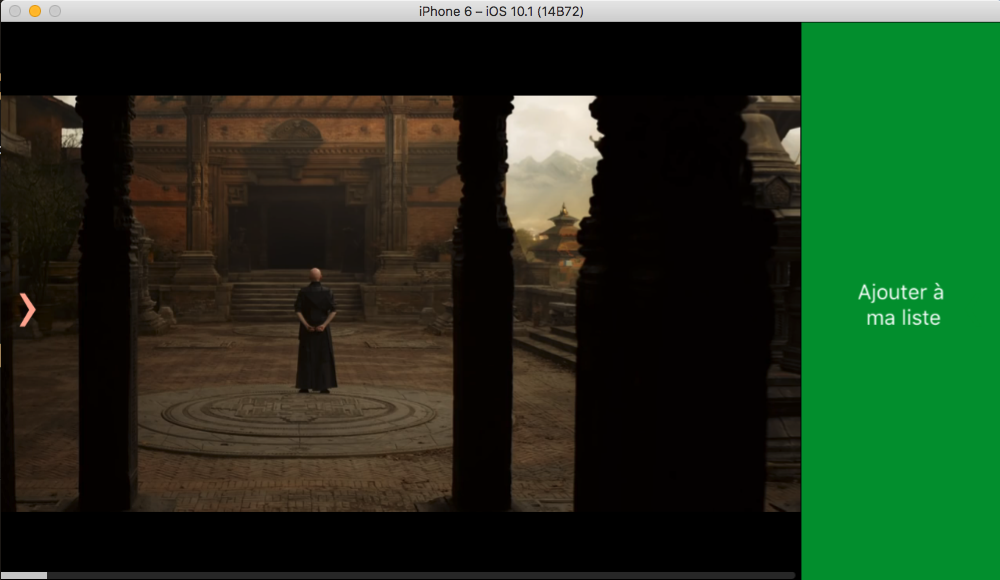
# Exécutable, sources et documentation

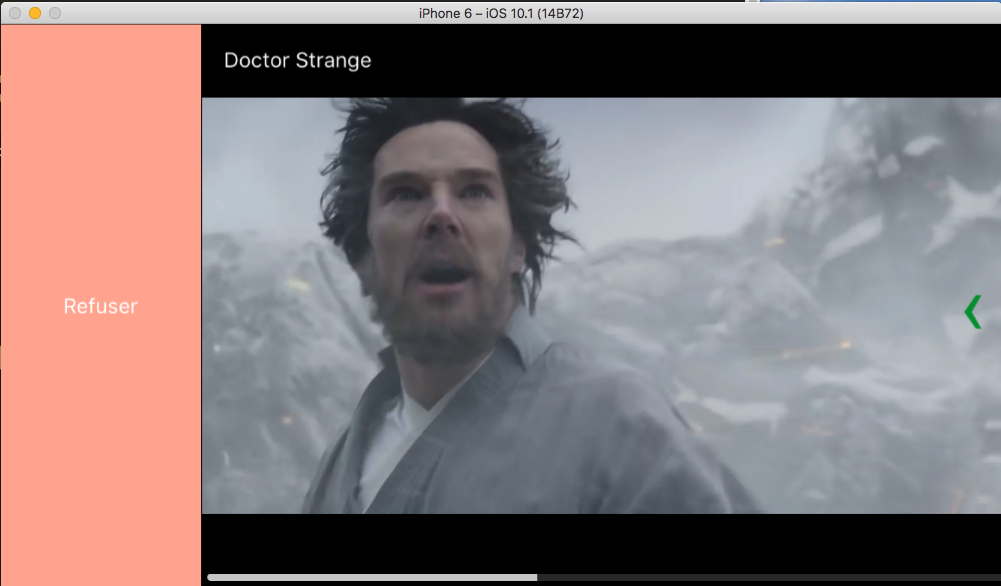
Guide d’utilisation :

La liste contient les films sélectionnés par l’utilisateur dans le module de visionnement accessible à partir de la page d’accueil. Cette liste est crée grâce au component « [ListView](https://facebook.github.io/react-native/docs/listview.html) » de React Native. Sur le clic d’un film, le navigateur de React Native nous amène dans une autre page pour consulter les détails du film choisi.



Le module de visionnement des vidéos est très épuré. Le film est affiché en plein écran et le titre est situé dans le haut. Une barre de progression personnalisée se trouve au bas de l’écran. Cette page est possible grâce à l’outil [React-native-youtube](https://github.com/inProgress-team/react-native-youtube) .





Pour ajouter ou refuser le film en cours, il suffit de balayer vers la droite ou la gauche.

Source et exécutable :

https://github.com/CharlesMcD/WatchNext

1. Drifty, « Build Amazing Native Apps and Progressive Web Apps with Ionic Framework and Angular ». [↑](#endnote-ref-1)
2. « AngularJS — Superheroic JavaScript MVW Framework ». [↑](#endnote-ref-2)
3. « Apache Cordova ». [↑](#endnote-ref-3)
4. « Node.js ». [↑](#endnote-ref-4)
5. « npm ». [↑](#endnote-ref-5)
6. « ng-video ». [↑](#endnote-ref-6)
7. « React Native | A framework for building native apps using React ». [↑](#endnote-ref-7)
8. « inProgress-team/react-native-youtube ». [↑](#endnote-ref-8)
9. « dancormier/react-native-swipeout ». [↑](#endnote-ref-9)
10. Nous avons choisi cette option lors du développement qui a beaucoup facilité la tâche grâce à la rapidité d’exécution d’une machine physique [↑](#footnote-ref-1)