

# Mode opératoire de test en local de l'Azure function

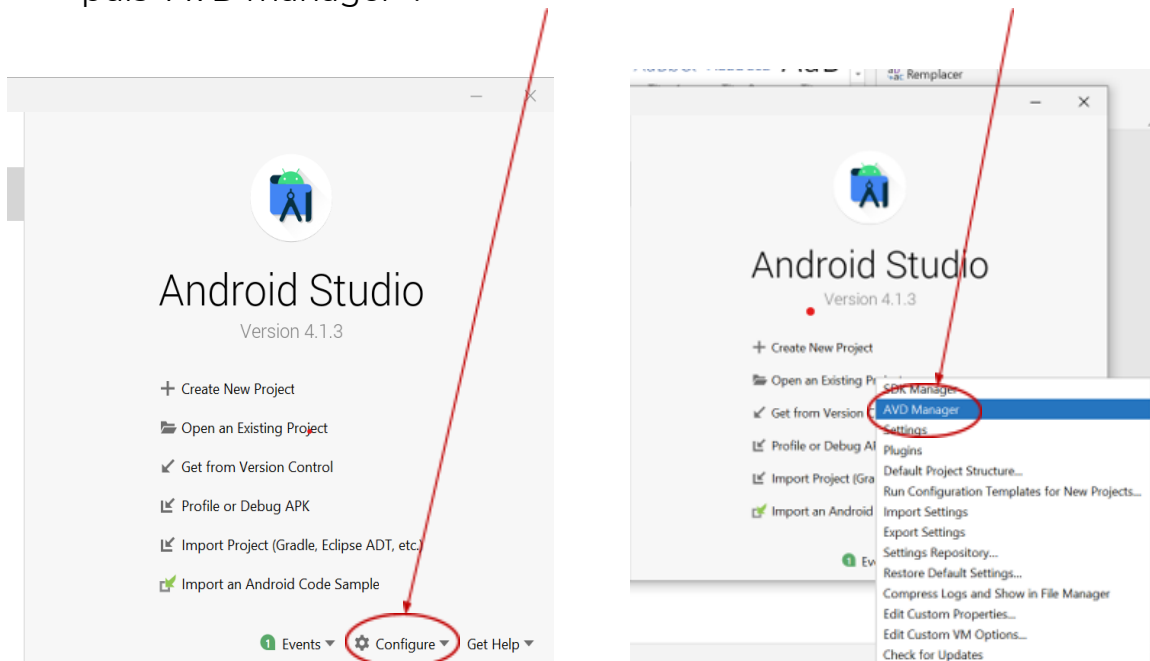
Ce document présente une démarche de mise en oeuvre de test en local d'une Azure function à partir de l'application Bookshelf exécutée à l'aide d'un émulateur Android, selon les étapes suivantes :

1. Installer et paramétrer Android Studio
2. Créer les répertoires en local
3. Préparer "bookshelf"
4. Installer et paramétrer VS Code pour Azure function
5. Tester l'Azure function en local
6. Tester l'Azure function en production

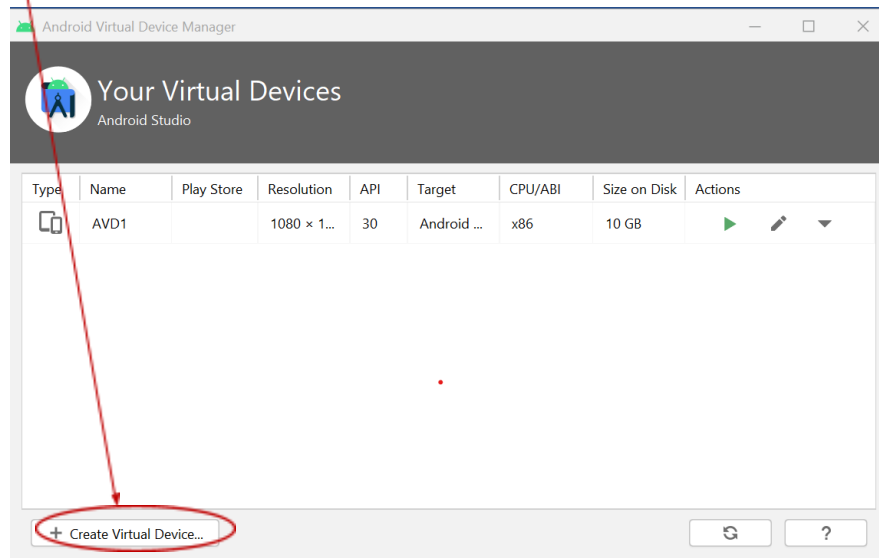
## 1. Installer et paramétrer Android Studio

Android Studio permet de simuler sur un PC Windows l'exécution d'un programme Android (par exemple Bookshelf), sans utilisation de smartphone ni de tablette :

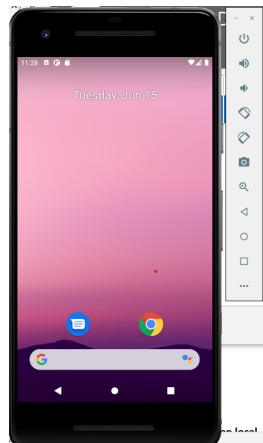
- Télécharger Android Studio via le lien : <https://developer.android.com/studio?hl=fr>, et l'installer sur son PC.
- Au lancement d'Android Studio, choisir "configure" en bas de l'écran, puis "AVD manager" :



- Créer un virtual device, avec de préférence une target Android 11.0



- Double cliquer sur le virtual device dans la liste (ici AVD1) pour ouvrir la simulation du smartphone. Un smartphone s'affiche :



## 2. Créer les répertoires en local

- Créer un répertoire "p9"
- Créer 2 sous-répertoires ("bookshelf", "functionOC")
- Créer un environnement virtuel « .venv » dans functionOC (par exemple commande « `virtualenv -p python3 .venv` »). L'installation des librairies via « requirements.txt » se fera automatiquement lors du test de la fonction en local (touche F5) ou lors du déploiement en production (CF étapes suivantes)



### 3. Installer et préparer l'application "bookshelf"

- Télécharger et installer la dernière mise à jour de Node.js : <https://nodejs.org/en/>, qui permet également de télécharger npm, intégré à ce package.
- Copier le code de Bookshelf de github dans le répertoire "p9/bookshelf".
- Dans ce répertoire "bookshelf", lancer en ligne de commande « npm install »
- Réaliser une correction dans « \bookshelf\node\_modules\metro-config\src\defaults\blacklist.js » = modifier le contenu de la variable « var sharedBlacklist » (ajout de plusieurs « \ ») :

```
[/node_modules[V\]react[V\]dist[V\].*/, /website\node_modulesV.*/, /heapCapture\bundle\.js/,  
/.*V__tests__V.*/];
```

(CF

<https://stackoverflow.com/questions/58268958/i-am-getting-invalid-regular-expression-error-while-running-npm-start>)

### 4. Installer et paramétrer VS Code pour Azure function

- Installer les plugins Azure si ce n'est pas déjà réalisé
- Installer Azure Functions Core Tools
- Dans VS Code se positionner sur le répertoire p9 (menu « File » puis « Open Folder »)
- Se connecter à une souscription
- Créer dans VS Code un projet local d'Azure function associé au répertoire déjà créé « p9/ functionOC »
- Dans VS Code, accéder au répertoire de travail "p9/functionOC"
- Adapter le code généré de l'Azure function, en particulier le programme « \_\_init\_\_.py » de l'Azure function et le fichier « requirements.txt »

Ressources :

- Show me Serverless (Guide vidéo Microsoft Azure function avec VS code) : [https://www.youtube.com/watch?v=coT4lIGQLCw&list=PLbl2SbVli-Wo2W81Jyqlv5B375W\\_EcUsj](https://www.youtube.com/watch?v=coT4lIGQLCw&list=PLbl2SbVli-Wo2W81Jyqlv5B375W_EcUsj)
- Tutoriel création Azure function (Créer une fonction dans Azure avec VS Code) : <https://docs.microsoft.com/fr-fr/azure/azure-functions/create-first-function-vs-code-python>



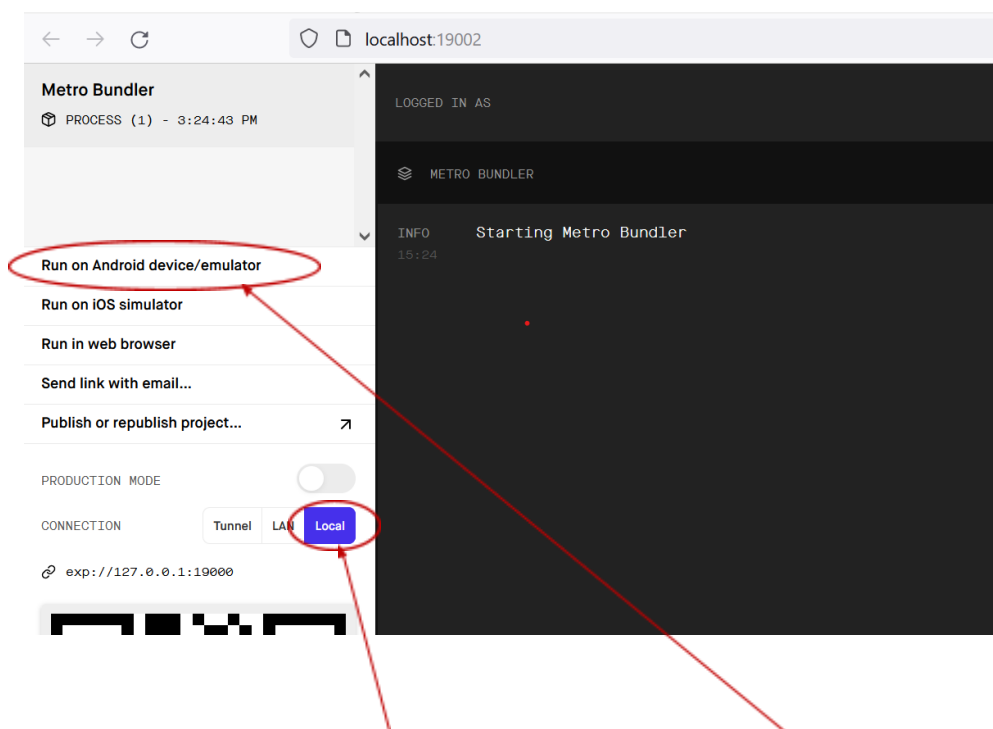
## 5. Tester l'Azure function en local

Dans le fichier config.json de bookshelf saisir le lien uri pour l'appel à l'Azure function (si nom du trigger = HttpTrigger1) : `{ "API_URL": "http://10.0.2.2:7071/api/HttpTrigger1" }`

Il est supposé que le point 1 a été réalisé, et que le virtual device est lancé (émulation smartphone affichée).

Dans VS Code, ouvrir un "terminal" et se placer dans le répertoire bookshelf :

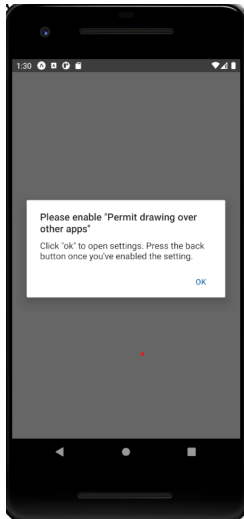
- Lancer npm install (si pas déjà fait)
- Lancer npm start, une page web s'affiche



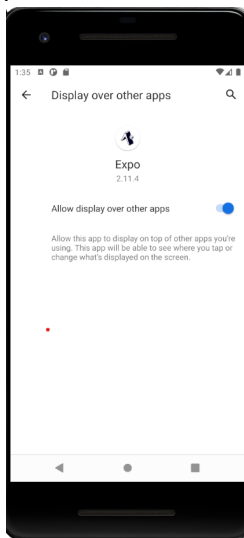
Sur cette page web choisir "Local" et cliquer sur "run on android device / emulator"

Dans le "terminal" de VS Code taper "y" s'il y a une demande d'upgrade d'Expo Go

Dans le simulateur cliquer sur OK pour "permit ..." si demandé



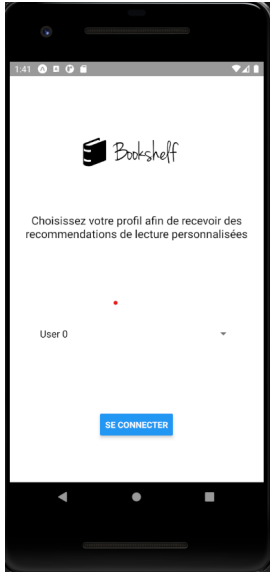
puis aller sur "expo" et cliquer sur "Allow ..." (curseur à droite)



puis cliquer sur flèche arrière 2 fois pour sortir de la gestion des permissions.

Dans VS Code taper F5 pour lancer le mode debug, qui lance notamment une installation de l'environnement (requirements.txt), dont l'exécution peut être suivie dans le « terminal » de VS Code. Penser à créer un environnement au préalable (CF étape 2)

L'application bookshelf s'affiche sur l'émulateur



Si l'application bookshelf ne s'affiche pas, éventuellement cliquer à nouveau sur "run on android device / emulator" (et si plantage de bookshelf cliquer à nouveau)

Tester Bookshelf



## 6. Tester l'Azure function en production

Après déploiement de l'Azure function en production, la démarche de test en production est similaire à celle au test en local :

- Dans le fichier config.json de bookshelf saisir le lien uri pour l'appel en production à l'Azure function (à récupérer sur le portail Azure) = `{ "API_URL": "uri_de_la_fonction_azure_créée" }`
- Lancer Android Studio et un virtual device (CF point 1)
- Dans VS Code, ouvrir un "terminal" et se placer dans le répertoire bookshelf
- Lancer npm install (si pas déjà fait)
- Lancer npm start, une page web s'affiche
- Sur cette page web laisser à "LAN" et cliquer sur "run on android device / emulator"
- Dans le "terminal" taper "n" s'il y a une demande d'upgrade d'Expo GO (en cas de problème, tester avec « y »)
- Eventuellement cliquer à nouveau sur "run on android device / emulator" (et si plantage cliquer à nouveau)
- L'application bookshelf s'affiche sur l'émulateur
- Tester Bookshelf