MANUEL D'INSTALLATION

1 Récupération des sources :

Les sources de la plateforme sont accessibles directement via GitHub via cette URL : https://github.com/aclaude1/plateformePC. Est présent dans ce répertoire :

1. Les sources Java du serveur
2. Les sources du Front-End
3. Un exemple de driver de test

2 Configuration de Docker

La plateforme utilise Docker pour pouvoir fonctionner. La première chose à faire est d'installer de de configurer Docker sur la machine host.

2.1 Installation de Docker

La documentation officielle de Docker présente comment installer le démon Docker sur une machine, quelque soit son OS (Linux, Mac-OS, et même Windows). Cependant, nous devons installer la plateforme sur une machine Linux, il faut donc aller sur cette URL : https://docs.docker.com/engine/installation/linux/.
Comme expliqué dans la documentation Docker, une simple commande permet de savoir très rapidement si l'installation a correctement été réalisée :

```
$ sudo docker run hello-world
```

Cette commande télécharge depuis le DockerHub la petite image *hello-world :lastest* et lance un composant ce cette image. Si l'installation à correctement été réussie, un petit message d'accueil apparaît.

2.2 Configuration de l'utilisation de docker

Lors de l'utilisation de toutes commandes Docker, le démon docker tournant sur la machine est contacté. Par défaut, seul le super utilisateur est autorisé à le contacté, et donc à exécuter des commandes docker. De ce fait, toutes commandes *docker* doivent être utilisées avec *sudo*, sinon, les requêtes n'aboutiront pas.
La plateforme ayant besoin de contacter le démon docker pour pouvoir fonctionner, deux solutions sont envisageables :

1. Le lancement de la plateforme sera lui aussi effectué en mode root.
2. On ajoute un utilisateur pour le management de docker. Cette manipulation autorise un utilisateur lambda à contacté le démon Docker, et donc nous n'aurons plus besoin de lancer la plateforme avec le mot-clef *root*. Les détails de cette manipulation sont expliqués ici : https://docs.docker.com/engine/installation/linux/linux-postinstall/#/manage-docker-as-a-non-root-user.

2.3 Création de l'image docker *alexandreinsa/base-plateforme*

La plateforme a besoin de l'image alexandreinsa/base-plateforme : elle sert de base pour construire les images que l'on crée lors de la création d'un nouvel exercice.
Cette image docker est stockée sur le DockerHub, c'est pourquoi il est très facile de la récupérer :

```
$ sudo docker pull alexandreinsa/base-plateforme
```

Au besoin, le fichier Dockerfile de cette image est disponible dans le répertoire /base-plateforme (où / est la base du répertoire du git). Une fois dans ce répertoire, il suffit de demander la création de l'image grâce au fichier Dockerfile :

```
$ sudo docker build -t alexandreinsa/base-plateforme .
```

Note : Le point est important et fait partie de la commande.
On peut vérifier la présence de l'image grâce à :

```
$ sudo docker images
```

3 La base MongoDB

3.1 Récupération de l'image Docker

La base mongoDB peut facilement être installée grâce à l'outil Docker, il suffit de récupérer la bonne image :

```
$ sudo docker pull mongo
```

3.2 Lancement de la base

Le lancement de la base se fait donc via Docker La base mongoDB peut facilement être installée grâce à l'outil Docker, il suffit de récupérer la bonne image :

```
$ sudo docker -d --name plateformeDB [-v </RepertoireHote>:/data/db]
-p 27017:27017 mongo
```

Quelques précisions :

1. Le nom de la plateforme (ici plateformeDB) est arbitraire, mais il faut s'en souvenir dans le cas où on aurait besoin de relancer le composant, ou d'aller directement voir dans la base de données. *Note : il peut facilement être retrouvé grâce à $ sudo docker ps -a.*
2. L'option facultative -v permet de lier les données de la base de données à un répertoire dans le système hôte. C'est pratique si par exemple on a prévu de migrer les données vers un autre système physique.
   Note : il faut que le répertoire hôte soit indiqué via un chemin absolu. Pour plus de précision, se référer ici

La base est maintenant lancée et le serveur saura s'y connecter de manière automatique, tout en créant les différentes COLLECTIONS au besoin. Si jamais la machine hôte tombe en panne, il suffit de relancer le composant, les données seront conservées.

```
$ sudo docker start plateformeDB
```

3.3 Lignes de commandes

Il est utile de connaître quelques commandes de shell MongoDB, au cas où on aurait besoin d'aller vérifier directement les données.
Connexion à la base de donnée :

```
$ sudo docker exec -it plateformeDB mongo
```

Utilisation de la base de donnée "plateforme" (utilisée par le serveur)

```
> use plateforme
```

Le serveur utilise la COLLECTION "Exercices", pour récupérer tous les exercices présents dans la base, on fait donc :

```
> db.Exercices.find()
```

Pour rechercher par exemple l'exercice ayant pour nom d'utilisation côté serveur (nom unique d'utilisation côté serveur) "Hello_World", on fait :

```
> db.Exercices.find({nomExoRepertoire : "Hello_World"})
```

4 Lancement de la plateforme

Une fois ces préparatifs réalisés, il ne reste plus qu'à lancer la plateforme. Avec VertX, le plus simple est de lancer un JAR, qui est facile via Gradle :

```
$ sudo ./gradlew runShadow
```

La page d'accueil sera directement accessible depuis un navigateur à l'URL http://localhost:8080/ressources/index.html.