Langages et Environements Évolués
TP0: Install party

26 septembre 2018

Avant toute installation, assurez-vous d'avoir assez de place. Pour les installation d'aujourd'hui, 2 Go (+1 Go de cache) devraient être suffisant, mais pour plus de sûreté, essayez de réserver 5Go pour l'ensemble du cours de LEE.

Il est particulièrement recommandé d'utiliser une distribution Linux. Il est aussi possible de faire tourner tous ces programmes sur Windows, MacOS ou BSD ; mais il est probable qu'il y ai des options différentes ou des soucis d'inter-opérabilité entre les logiciels, en particulier pour Docker (nous y reviendront).

Exercice 1 (Java EE).
Terminologie officielle :

- Java SE (Standard Edition) : noyau des API (lang, collections, ..., jar, ...) sans machine virtuelle.
- JRE (Java Runtime Environment) : JSE + machine virtuelle + plugin.
- JDK (Java Development Toolkit) : JRE + logiciels d'utilisation (java, javac, jar, ...).
- JEE/J2EE (Java Enterprise Edition) : JDK + Serveur d'application (glasfish, wildFly...)

Attention, pour des raisons de com, oracle ne respecte pas sa propre terminologie sur le site-web et un téléchargement d'un JDK peut être indiqué comme un JRE ou un téléchargement de glassfish peut être indiqué comme un téléchargement de JEE...
Installation :

1. Téléchargez JDK 10 sur le site d'Oracle.
2. Téléchargez Java EE 8 sur le site d'Oracle. (En fait, il n'y a pas grand chose de plus que le serveur Glassfish.)

Exercice 2 (IntelliJ).
IntelliJ est l'environnement de développement intégré (IDE) que l'on va utiliser tout le semestre. Il intègre l'utilisation de tous les outils que nous verrons en TP (Java, Java EE, hibernate, Spring, AngularJS, Docker, Git) et d'autres encore (Scala, GRails, Android tools, Maven, Groovy, Kotlin, etc...).

1. Si vous ne l'avez pas déjà fait, inscrivez-vous comme étudiant sur le site de JetBrain à l'aide de votre adresse mail `@univ-Paris13.fr`.
2. Téléchargez IntelliJ Ultimate.
3. Décompressez le premier et placez y un terminal puis exécutez `./bin/idea.sh`, cela débutera l'installation.
4. Entrez vos identifiants JetBrain.
5. Importez vos configs si vous aviez déjà une version publique d'IntelliJ installée.
6. Choisissez l'apparence que vous préférez.
7. Il faut maintenant choisir les plugin : dans le première fenêtre, il n'y a que les outils "Build Tools" et "Android" que l'on ne verra pas (principalement par manque de temps), vous pouvez les décocher si vous le souhaitez.
8. Dans la seconde fenêtre, il vous faudra récupérer le plugin Angular (intervenant pro). Les élèves de PLS souhaiteront aussi récupérer Scala (pour le cours du second semestre). Vous pouvez, bien sur en télécharger d'autres s'ils vous intéressent.

Exercice 3 (Docker). Docker est un outil de virtualisation légère. Il a deux utilités principales : en phase de développement il permet de fournir un environnement de tests uniforme et réaliste pour vos programmes, en phase de déploiement il permet un déploiement léger, distribué et optimisable (mais est difficile à paramétrer et à sécuriser).

1. Sur linux : essayez de taper `sudo apt-get innstall docker`. Si cela fonctionne (testé sur debian testing seulement) sautez la prochaine étape. [1]
2. Suivez les instruction disponible sur le site officiel Docker pour utiliser la version libre "community edition" (CE). Ces instructions dépendent fortement de votre distribution car Docker interagit directement avec le noyau. Attention : vous ne pourrez utiliser Docker que avec des droits root, autrement le Daemon ne pourra pas se lancer.
3. Installez Docker-compose
4. Testez dans le terminal `sudo docker version` et `sudo docker-compose version`.
5. Il faut maintenant intégrer Docker à IntelliJ : [2]
   Dans IntelliJ, faites `Ctrl+Alt+S` pour acceder aux `Settings` (vous pouvez aussi y acceder via les menus déroulants). Allez dans plugins et recherchez `Docker integration`, installez-le.
   Réouvrez les `Settings` et allez dans `Build,[...]`→`Docker`. Dans la seconde column, cliquez sur le + et vérifiez que vous êtes bien connecté au démon Docker avec la mention `Connection successful` dans la 3ieme colone. Si ce n'est pas le cas, c'est que vous avec un soucis avec Docker.
6. Pour le `TP1_team`, il est conseillé de télécharger l'image `wildfly` en avance, car il s'agit de 200M... Pour celà, attendez d'avoir une connection raisonable et tapez dans le terminal :

   ```
   sudo docker pull jboss/wildfly
   ```

Exercice 4 (Docker sous IntelliJ [3]). On souhaiterait maintenant intégrer Docker à IntelliJ. Sous linux, vous allez avoir un soucis car IntelliJ est idéalement lancé en tant qu'utilisateur et Docker en tant que root, si on veut déléguer la gestion de docker à IntelliJ, on va avoir un soucis de droits. Il y a deux solutions au problèmes (toutes deux médiocres...) : la première est de lancer IntelliJ en sudoer, la seconde est de donner les droit docker à votre utilisateur (les droits docker sont équivalents au droit root, mais il faut passer par docker ce qui ralentie les attaques). Entre la peste et le choléra, [4] je conseillerais plutôt la seconde pour les utilisateur linux natif, et la première pour ceux qui utilisent une machine virtuelle. Ceux qui choisissent la première option et les utilisateurs de Windows peuvent sauter la première étape.

1. Ajout de l'utilisateur au groupe Docker. [5]
   Lancez les deux commandes suivantes dans un terminal :

   ```
   $ sudo groupadd docker
   $ sudo usermod -aG docker $USER
   ```

   redémarrez votre session (logout), puis testez si Docker fonctionne sans sudo :

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

   S'il y a un message d'erreur, tapez ces deux commandes :

   ```
   $ sudo chown "$USER":"$USER" /home/"$USER"/.docker -R
   $ sudo chmod g+rwx "/home/$USER/.docker" -R
   ```

   redémarrez votre session (logout), puis testez Docker.
2. Dans IntelliJ, faites `Ctrl+Alt+S` pour accéder aux `Settings` (vous pouvez aussi y accéder via les menus déroulants). Allez dans plugins et recherchez `Docker integration`, installez-le.
   Réouvrez les `Settings` et allez dans `Build,[...]`→`Docker`. Dans la seconde colonne, cliquez sur le + et vérifiez que vous êtes bien connecté au démon Docker avec la mention `Connection successful` dans la 3ieme colonne. Si ce n'est pas le cas, c'est que vous avec un soucis avec Docker.

1. Le package docker-compose existe lui aussi, mais il est pour l'instant buggé.
2. Repris de la doc IntelliJ.
3. Repris de la doc IntelliJ.
4. On peut espérer de voir arriver, dans les années à venir, un droit docker restreint avec un sous-ensemble raisonnable des commandes Docker...
5. repris du site officiel Docker

Exercice 5 (Git). Git est un gestionnaire de versions, il permet de de sauvegarder votre code, de garder des versions précédentes et de travailler à plusieurs sur un même code ; tout cela de manière efficace et avec un minimum de conflits (on le reverra en cours).

1. Installez git :
   — Sous linux :
   `$ sudo dnf install git-all`
   — Sous Windows :
   Téléchargez l'exécutable et installez le.
   — Sous Mac :
   Téléchargez l'exécutable et installez le.
   — Si vous n'avez pas encore de compte github/gitlab, inscrivez-vous sur github. [6] A savoir : mettre vos projets sur un github (ou un autre host publique) permet à vos recruteurs de se renseigner sur ce que vous avez pu coder, dans les cours ou en dehors, et sur votre qualité de code, c'est extrêmement profitable pour l'embauche.

6. Vous pouvez préférer un autre host de serveurs git, surtout maintenant que github a été racheté par Microsoft ; sachez simplement que github est encore le plus utilisé...