Dossier d’installation

Ce document est destiné aux développeurs qui souhaitent mettre en place un environnement de développement.

**Thomas GIRAULT**

Dossier d’installation de l’environnement de développement v1.0

# Table des illustrations

**Aucune entrée de table d'illustration n'a été trouvée.**

Sommaire

[Table des illustrations 1](#_Toc446871432)

[Suivi des modifications 3](#_Toc446871433)

[Caracteristiques techniques 3](#_Toc446871434)

[1. Prérequis 4](#_Toc446871435)

[1.1. Caractéristiques de la machine 4](#_Toc446871436)

[1.2. Configuration réseau 4](#_Toc446871437)

[1.2.1. Configuration de l’interface réseau 4](#_Toc446871438)

[1.2.2. Configuration des paramètres réseau globaux 4](#_Toc446871439)

[1.2.3. Configuration du nom d’hôte de la machine 4](#_Toc446871440)

[1.2.4. Configuration du FQDN de la machine 5](#_Toc446871441)

[1.2.5. Configuration des serveurs DNS 5](#_Toc446871442)

[1.2.6. Redémarrage du service réseau 5](#_Toc446871443)

[1.1. Configuration des 5](#_Toc446871444)

[2. Installation de docker 5](#_Toc446871445)

[3. Jenkins 5](#_Toc446871446)

[3.1. Démarrer Jenkins 5](#_Toc446871447)

[4. Shipyard 6](#_Toc446871448)

[4.1. Démarrer Shipyard 6](#_Toc446871449)

[4.1.1. Datastore 6](#_Toc446871450)

[4.1.2. Discovery 6](#_Toc446871451)

[4.1.3. Proxy 6](#_Toc446871452)

[4.1.4. Swarm Manager 7](#_Toc446871453)

[4.1.5. Swarm Agent 7](#_Toc446871454)

[4.1.6. Controller 7](#_Toc446871455)

[4.2. Arrêter Shipyard 8](#_Toc446871456)

[4.2.1. Arrêter les conteneurs Shipyard 8](#_Toc446871457)

[4.2.2. Supprimer les conteneurs Shipyard 8](#_Toc446871458)

[5. Installer un serveur Git 8](#_Toc446871459)

[6. Annexes 9](#_Toc446871460)

[6.1. Variables d’environnement 9](#_Toc446871461)

# Suivi des modifications

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Description | Modifié par | Version |
| 12/01/2018 | Initialisation du document | TGI | v1.0 |
|  |  |  |  |
|  |  |  |  |

# Prérequis

## Caractéristiques de la machine

Création d’une machine virtuelle avec Hyper-V :

|  |  |
| --- | --- |
| Caractéristiques | |
| Système d’exploitation | Linux Centos 7 |
| Type du système | Système d’exploitation 64 bits |
| Processeur | Intel® Core 2 Duo CPU P8700 @ 2.53 GHz 2.53 GHz |
| Mémoire installée (RAM) | 4,00 Go |
| Espace disque | 10 Go |

## Configuration réseau

Se connecter sur le serveur avec le compte administrateur root.

### Configuration de l’interface réseau

1. Éditer le script de l’interface réseau /etc/sysconfig/network-scripts/ifcfg-eth0
2. Renseigner les lignes suivantes :

TYPE=Ethernet

BOOTPROTO=static

NAME=eth0

DEVICE=eth0

ONBOOT=yes

HWADDR=AA:BB:CC:DD:EE:FF

IPADDR=192.168.0.100

NETWORK=192.168.0.0

NETMASK=255.255.255.0

BROADCAST=192.168.0.255

GATEWAY=192.168.0.1

DNS1=192.168.0.100

DNS2=8.8.8.8

### Configuration des paramètres réseau globaux

1. Éditer le fichier /etc/sysconfig/network
2. Renseigner les lignes suivantes :

NETWORKING=yes

NETWORKING\_IPV6=no

### Configuration du nom d’hôte de la machine

1. Éditer le fichier /etc/hostname
2. Renseigner les lignes suivantes :

HOSTNAME=slnxmasterint01

Le nom d’hôte suit le format slnx<projet><env><numero>.

### Configuration du FQDN de la machine

1. Éditer le fichier /etc/hosts
2. Renseigner les lignes suivantes :

192.168.0.100 slnxmasterint01.fr.local slnxmasterint01

FQDN : Fully Qualified Domain Name

### Configuration des serveurs DNS (facultatif)

1. Éditer le fichier /etc/resolv.conf
2. Renseigner les lignes suivantes :

nameserver 8.8.8.8

nameserver 8.8.4.4

### Redémarrage du service réseau

1. Redémarrer le service réseau

service network restart

# Installation de docker

Installation de Docker Community Edition (CE)

1. Installation des pré-requis

sudo yum install -y yum-utils device-mapper-persistent-data lvm2

1. Configuration du repository docker-ce

sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

1. Installation de docker-ce

sudo yum install docker-ce

1. Démarrage de docker

sudo systemctl start docker

1. Configurer le démarrage automatique du service au démarrage du système

chkconfig docker on

Source : <https://docs.docker.com/engine/installation/linux/docker-ce/centos/#install-docker-ce-1> (12/01/2018)

# Installation de docker compose

1. Télécharger la dernière version de docker compose

sudo curl -L https://github.com/docker/compose/releases/download/1.18.0/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose

1. Ajout des droits d’exécution sur le binaire

sudo chmod +x /usr/local/bin/docker-compose

1. Tester l’installation

[root@slnxcentos ~]# docker-compose --version

docker-compose version 1.18.0, build 8dd22a9

# Shipyard

Shipyard est un outil qui permet d’administrer les conteneurs Docker.

## Démarrer Shipyard

### Démarrage automatique

curl -sSL https://shipyard-project.com/deploy | bash -s

### Démarrage manuel

### Datastore

Démarre le conteneur RethinkDB, la base de données utilisée par Shipyard.

docker run -ti -d --restart=always --name shipyard-rethinkdb rethinkdb

### Discovery

Pour activer l’élection d’un maître Swarm, nous devons utiliser une paire clé/valeur externe stockée dans le conteneur Swarm. Dans l’exemple qui suit, nous utilisons etcd. Cependant vous pouvez utiliser n’importe quelle clé/valeur supportée par Swarm.

docker run -ti -d -p 4001:4001 -p 7001:7001 --restart=always --name shipyard-discovery microbox/etcd -name discovery

### Proxy

Démarre le conteneur proxy qui redirige simplement les requêtes TCP vers le socket Unix sur lequel Docker écoute.

docker run -ti -d -p 2375:2375 --hostname=$HOSTNAME --restart=always --name shipyard-proxy -v /var/run/docker.sock:/var/run/docker.sock -e PORT=2375 shipyard/docker-proxy:latest

### Swarm Manager

Démarre un conteneur Swarm configuré pour gérer les contenurs.

docker run -ti -d --restart=always --name shipyard-swarm-manager swarm:latest manage --host tcp://0.0.0.0:3375 etcd://192.168.0.100:4001

### Swarm Agent

Démarre l’agent Swarm qui autorise les nœuds à ordonnancer les conteneurs.

docker run -ti -d --restart=always --name shipyard-swarm-agent swarm:latest join --addr 192.168.0.100:2375 etcd://192.168.0.100:4001

### Controller

Démarre le contrôleur Shipyard.

docker run -ti -d --restart=always --name shipyard-controller --link shipyard-rethinkdb:rethinkdb --link shipyard-swarm-manager:swarm -p 8080:8080 shipyard/shipyard:latest server -d tcp://swarm:3375

Dès lors que le contrôleur est démarré, il est possible de se connecter à l’adresse suivante : http://192.168.0.100:8080

* Login : admin
* Mot de passe : shipyard

Lien: http://shipyard-project.com/docs/deploy/manual/

## Arrêter Shipyard

Arrêter les conteneurs Shipyard.

docker stop shipyard-controller

docker stop shipyard-swarm-agent

docker stop shipyard-swarm-manager

docker stop shipyard-proxy

docker stop shipyard-certs

docker stop shipyard-discovery

docker stop shipyard-rethinkdb

Supprimer les processus Shipyard.

docker ps -a | grep 'shipyard' | awk '{print $1}' | xargs --no-run-if-empty docker rm

# Jenkins

## Démarrer Jenkins

Démarrer le conteneur Jenkins.

docker run --name jenkins -d -p 8081:8080 -p 50000:50000 -v /var/jenkins\_home jenkins

**Paramètres:**

-d : permet d’exécuter le conteneur en tâche de fond

-p : permet d’associer les ports réseau du conteneur avec les ports de la machine hôte.

-v : permet d’associer un volume

Dès lors que le contrôleur est démarré, il est possible de se connecter à l’adresse suivante : http://192.168.0.100:8081

Pour récupérer le mot de passe initial, taper la commande :

docker exec jenkins cat /var/jenkins\_home/secrets/initialAdminPassword

* Login : admin
* Mot de passe : admin

Lien: https://hub.docker.com/\_/jenkins/

## Arrêter Jenkins

Arrêter le conteneur Jenkins.

docker stop jenkins

Supprimer le processus jenkins.

docker ps -a | grep 'jenkins' | awk '{print $1}' | xargs --no-run-if-empty docker rm

# JHipster

## JHipster Registry

Télécharger l’image docker

docker pull jhipster/jhipster-registry

Démarrer l’image docker

docker run -d -p 8761:8761 -e JHIPSTER\_SECURITY\_AUTHENTICATION\_JWT\_SECRET=mysecretjwtkey --restart=always jhipster/jhipster-registry:latest

* Login : admin
* Mot de passe : admin

## JHipster microservice application

Installer JHipster

Lien : <http://www.jhipster.tech/installation/> (12/01/2018)

1 VM slnxint01

* Docker
* Jenkins
* Sonar
* Shipyard

1 VM slnxdev01

* Docker
* Jhipster
* Repository local Git
  + Push / pull on GitHub

Repository Git (Git Hub)

Repository Docker

# Annexes

## Variables d’environnement

|  |  |  |
| --- | --- | --- |
| Variable | Valeur | Description |
| $\_ IPADRESS | 192.168.0.100 | Adresse IP de la machine |
| $\_HOSTNAME | slnxmasterint01 | Nom d’hôte de la machine |
| $\_FQDN | slnxmasterint01.fr.local | Fully Qualified Domain Name de la machine |
| $\_DOMAIN |  | Domaine de la machine |
| $\_NETWORK | 192.168.0.0 | Adresse du réseau |
| $\_GATEWAY | 192.168.0.1 | Passerelle vers le réseau |
| $\_NETMASK | 255.255.255.0 | Masque du réseau |
| $\_BROADCAST | 192.168.0.255 | Adresse de broadcast |
| $\_DNS1 | 8.8.8.8 | Adresse DNS 1 |
| $\_DNS2 | 8.8.4.4 | Adresse DNS 2 |

## Commandes docker

* Exécuter l’invité de commande du conteneur

docker container exec -it <container-name> sh

* Consulter les processus actifs

docker ps -a

* Consulter les images téléchargées

docker image

## Gestion de version Git

### Pré-requis :

* Disposer d’un compte GitHub : <https://github.com/tgirault>
* Git est installé sur le poste de développement

### Créer un nouveau projet

1. Créer un nouveau repository sur GitHub.

Pour éviter des erreurs, ne pas initialiser un nouveau repository avec un README, une license, ou des fichiers gitignore. Ces fichiers pourrons être ajoutés après que le projet soit poussé vers GitHub

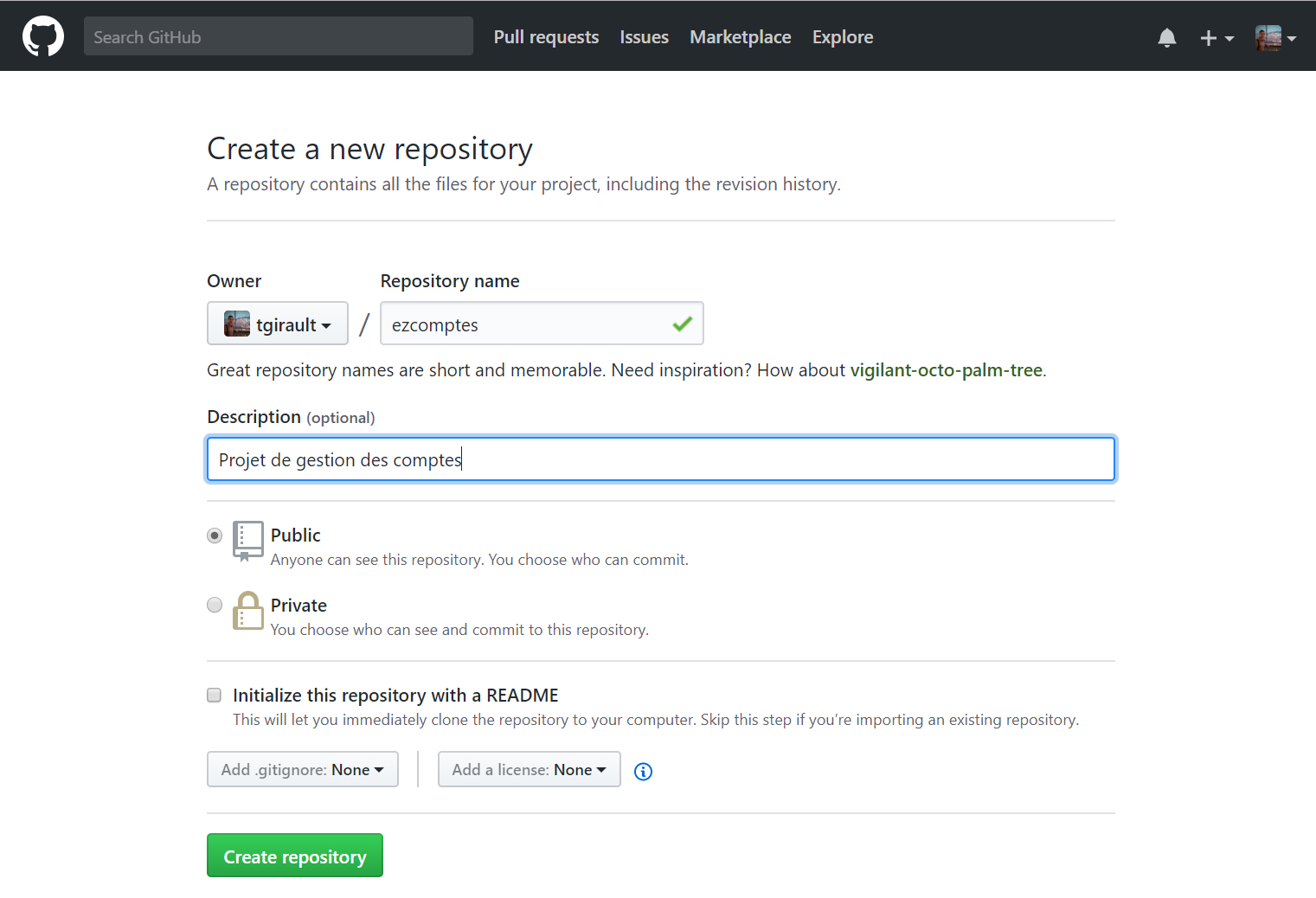


Figure 1 Création d'un repository GitHub

Nouveau repository : https://github.com/tgirault/ezcomptes.git

1. Ouvrir un invité de commandes dans le répertoire du projet local
2. Initialiser le répertoire local en tant que repository Git

git init

1. Ajouter les fichiers dans le nouveau repository local

git add .

# Adds the files in the local repository and stages them for commit. To unstage a file, use 'git reset HEAD YOUR-FILE'.

1. Commit les fichiers sélectionnés dans le repository local

git commit -m "First commit"

# Commits the tracked changes and prepares them to be pushed to a remote repository. To remove this commit and modify the file, use 'git reset --soft HEAD~1' and commit and add the file again.

1. Copier l’URL du repository distant en haut de la page du repository GitHub
   * https://github.com/tgirault/ezcomptes.git

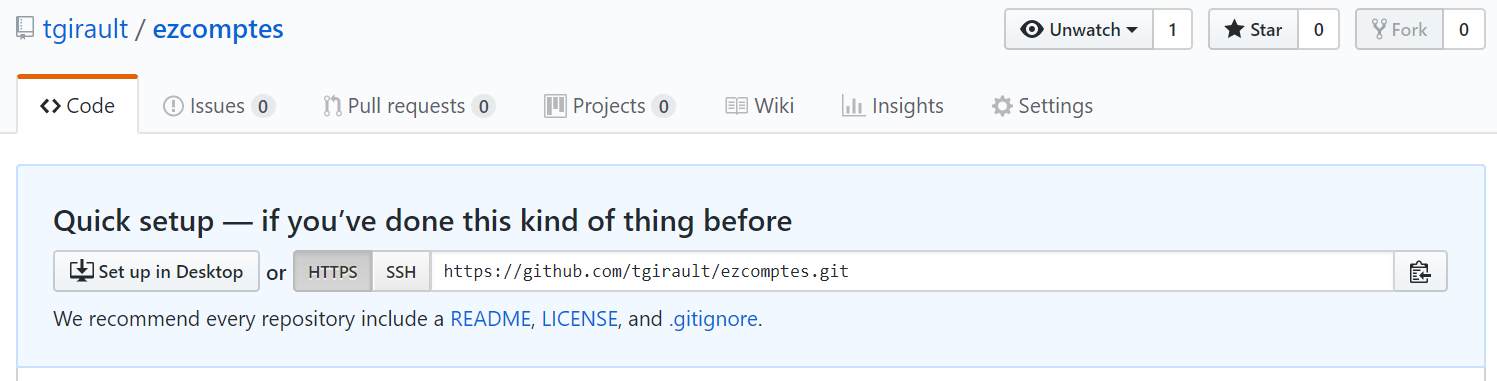


Figure 2 URL du repository GitHub

1. Dans l’invité de commandes, ajouter l’URL du repository distant vers lequel seront push les fichiers du repository local

git remote add origin remote repository <URL>

# Sets the new remote

git remote -v

# Verifies the new remote URL

1. Pousser les fichiers présents dans le repository local vers le repository GitHub

git push origin master

# Pushes the changes in your local repository up to the remote repository you specified as the origin

Source : <https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/>

### Récupérer les sources d’un projet GitHub