git-df

(*testé* - **smolski** 26/10/2012) Le retour sur le forum est ici :

- Lien vers le forum concernant ce tuto N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !

- Objet : git pour df
- Niveau requis : AVISÉ
- Commentaires : *Ce tuto est destiné essentiellement aux participants des projets df via le dépôt git installé sur le serveur df.*
- Débutant, à savoir :

Utiliser GNU/Linux en ligne de commande, tout commence là !.

Installation

Terminal root :

Mettre son système à jour :

```
apt-get update
apt-get upgrade
```

Installer git-core :

```
apt-get install git-core
```

Tout le reste du tuto se passe en terminal *utilisateur* sauf indication contraire.

Initialisation

Tapez :

```
git config --global user.name "votre pseudo du forum" Majuscules ou espace
autorisés.
git config --global user.email "votre@mail.com"
git config --global color.ui true
git config --global core.quotePath false
```

Ces commandes vont créer un fichier dans votre répertoire utilisateur qui sera utilisé par tout vos projets. Par défaut, ce fichier est ~/.gitconfig et il contiendra quelque chose comme ça :

```
    [user]
```

```
      name = votre pseudo du forum
      email = votre@mail.com
```

Ce *name* vous servira en signature de vos interventions et dans vos tag.

Clé SSH

Créer les clés sh

Créer une paire de clés ssh[1].

nom_ou_pseudo = Votre nom ou votre pseudo, sans espace ni accent et en minuscule[2] !

```
ssh-keygen -f ~/.ssh/df-git-nom_ou_pseudo
```

Il vous sera demandé si vous voulez ajouter une passphrase, à vous de voir si cela vous paraît nécessaire... Notez que si vous en saisissez une, il vous faudra la retaper à chaque nouvelle session de votre pc avant d'utiliser les dépôts.

Ce qui donnera au final dans le répertoire `~/.ssh` :

1. un fichier de clé privée[3]
2. et un de clé publique[4]

Vérifiez :

```
ls ~/.ssh
```

```
df-git-nom_ou_pseudo  df-git-nom_ou_pseudo.pub
```

Communiquez la clé publique

Il vous faudra communiquer au captnfab le fichier de la clé publique[5] **df-git-nom_ou_pseudo.pub**, par email ou depuis le forum df par mp par exemple...

Cette clé publique n'est pas confidentielle. Elle permet simplement au serveur de vérifier que c'est bien vous qui vous connectez avec votre clé privée !

Éditez ensuite le fichier `~/.ssh/config`, à créer s'il n'existe pas. Rajouter dans ce dernier le bloc de lignes que vous aura envoyé le captnfab en réponse à votre message.

Créez le répertoire contenant vos git-projets-df

Ensuite, créez un répertoire dans votre `/home/user` (c'est le lieu le plus simple) qui vous servira de répertoire où faire transiter vos travaux git mis en commun sur le git-df.
Par exemple nommé *projets-df* :

```
mkdir ~/projets-df
```

passphrase

Si vous avez créé une passphrase tapez :

```
ssh-add ~/.ssh/df-git-votre_pseudo
```

Cette ligne ajoute la passphrase en mémoire en la donnant à ssh-agent.

Cloner un projet déjà en cours

À l'intention des sombres personnes qui gisent habituellement au fond des classes près des radiateurs, précisons bien ici que git ne vous transite pas vous vers le serveur, mais transitent uniquement les données échangées entre votre pc vers le serveur git distant, et vice et versa. *Qu'on se le dise* ! (ce que jojo veux dire, c'est que vous n'avez pas de shell sur le serveur)

Afin de connaître la liste des dépôts auxquels vous avez accès en lecture ou en écriture, tapez la commande suivante :

```
ssh -T df-git
```

```
hello contributeur, the gitolite version here is 2.0.3
the gitolite config gives you the following access:
    @R_ @W_     bac-a-sable
    @R   W      docs/ebook-facile
    @R          outils/live-df
    @R   W      tutos/latex/pour-commencer
    @R          tutos/systeme/c-shell
```

Dans cet exemple, l'utilisateur “contributeur” a accès en écriture aux dépôts `bac-a-sable` (un dépôt pour faire des tests si vous n'êtes pas familiers avec git), `docs/ebook-facile`, le dépôt pour le projet ebook et `tutos/latex/pour-commencer`. Il n'a qu'un accès en lecture aux deux autres projets.

En tant que contributeur, vous avez accès en lecture/écriture aux projets auxquels vous participez, et en lecture seule aux autres projets. Il est en projet de faire en sorte

que les non-contributeurs aient un accès web en lecture seule à tous les projets.

Se placer dans ce répertoire créé pour git :

```
cd ~/projets-df
```

Demander le téléchargement du répertoire du projet par clonage ainsi :

```
git clone df-git:le_projet_commun
```

Vous aurez à disposition personnelle ce répertoire et tout ce qu'il contient mis dans le répertoire git d'où vous avez lancé la commande :

```
 ~/projets-df/le_projet_commun
```

Utilisation

Si vous préférez voir un cas typique d'utilisation avant de lire la liste des commandes, le TP1 suivant s'y prête bien :

- TP 1 : Exemple d'utilisation basique

Liste des commandes

Placez-vous dans le répertoire cloné du *fichier_travail*

```
cd ~/projets-df/le_projet_commun/
```

Vous pouvez modifier ou compléter le fichier_travail, par exemple avec vim :

```
vim fichier_travail
```

Lorsque vous avez fini et souhaitez partager ce que vous avez réalisé, vous l'ajoutez :

add

```
git add fichier_travail
```

commit

Vous le commitez :

```
git commit -a
```

Vous aurez alors accès à la rédaction de ce commit ainsi :

```
  1 Tests matutinaux d'utilisaton du git-df <- Ici une ligne vierge qui sert
au titrage référence de votre intervention dans le projet.
  2
  3 Ici est un endroit à ajouter si on le désire pour raconter un historique
qui serait trop long dans le titre du dessus.
  4 Bien respecter l'interligne vierge entre les deux.
  5 # Please enter the commit message for your changes. Lines starting
  6 # with '#' will be ignored, and an empty message aborts the commit.
  7 #
  8 # Committer: votre pseudo du forum. Votre mail.
  9 #
 10 # On branch master
 11 # Changes to be committed:
 12 #   (use "git reset HEAD <file>..." to unstage)
 13 #
 14 #       modified:   test
 15 #
```

En enregistrant tout cela dans votre terminal vous obtiendrez ensuite :

```
[master 4daaf19] Tests matutinaux d'utilisaton du git-df
 Committer: votre pseudo du forum. Votre mail.

If the identity used for this commit is wrong, you can fix it with:

    git commit --amend --author='Your Name <you@example.com>'

 1 files changed, 1 insertions(+), 1 deletions(-)
```

push

Vous le poussez vers le git-df :

```
git push
```

Vous obtiendrez alors l'édition de ces lignes :

```
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 432 bytes, done.
Total 3 (delta 0), reused 0 (delta 0)
To df-git-nom_pseudo:fichier_travail
   xxxxxxx..xxxxxxx  master -> master
Already up-to-date.
```

pull

Plus tard, pour le tirer de nouveau vers vous avec toutes les interventions ajoutées :

```
git pull
```

Et ainsi de suite...

Glossaire

1. commit : Dans un gestionnaire de suivi de version comme git, chaque contributeur, après avoir apporté des modifications de son côté, peut les ajouter au projet principal. Cela s'appelle faire un commit. Le gestionnaire de version permet, en cas de problème, de chercher parmis tous les commit la modification qui posait problème.
2. add : Cela sert à ajouter un nouveau fichier au dépot. Ce n'est pas une simple modification d'un fichier déja existant, comme avec commit.
3. push : (pousser) Cela sert à envoyer les modifications que vous avez apporté (les commits) sur le dépot principal. En cas de conflit avec des modifications d'un autre utilisateur, vous pouvez les résoudre à la main
4. pull : Cela vous permet d'être à jour avec le dépot, et les modifications apportées par les autres.

Création projet

À suivre, jeu sais pas encore faire ça... 😉

TP

1. TP 1 : Exemple d'utilisation basique
2. TP 2 : Résoudre des conflits

Lien et remerciement

Tuto référentiel :

- Bienvenue sur git - alexgirard

Merci à **enicar** *et au* **captnfab** 😉

1)
Configurer ssh
2)
En principe vous avez le droit mais, dans notre tuto et pour éviter les erreurs, on adoptera cette convention
3)
df-git-nom_ou_pseudo

4)

df-git-nom_ou_pseudo.pub

5)

ou son contenu