
Configurer et utiliser GIT à minima

Notions : Introduction à un gestionnaire de versions partagé

1 Principes de fonctionnement

Git permet de partager des fichiers de code ou autres entre un ou plusieurs programmeurs et assure la gestion des versions : toutes modification d'un fichier est réversible, vous pouvez toujours revenir à une version précédente de vos fichiers : c'est un peu le *Annuler* de vos logiciels habituels, mais cela peut aussi s'appliquer à des groupes de fichiers, à des répertoires, à des versions écrites il y a X années, etc....

Nous vous avons créé un dépôt **central** sur le gitlab gitlab.ensimag.fr qui va contenir tous vos fichiers et sujets de be. Vous allez pouvoir obtenir une copie de ces fichiers sur TOUTES les machines sur lesquelles vous travaillerez : celle de Phelma, mais aussi les vôtres. Sur chacune de ces machines, vous aurez un dépôt **local**. Si vous avez 3 machines, vous aurez donc 3 copies locales, qui seront toutes à jour si vous utilisez bien *git*. *git* sert à conserver et mettre à jour toutes les copies que vous avez.

Initialement, pour **chaque** machine sur laquelle vous souhaitez avoir une copie, il faut commencer par *cloner le dépôt global* i.e. créer le dépôt **local** à l'aide du dépôt **global**. Vous pouvez bien sûr créer un nouveau dépôt à n'importe quel moment, sur n'importe quelle machine : il sera parfaitement à jour avec l'état du dépôt global à ce moment.

Ensuite, à chaque fois que vous travaillez sur une machine, il faut :

1. commencer par mettre à jour le dépôt **local** car ce dépôt n'est peut-être pas à jour si vous ou un autre programmeur avez modifié le dépôt **global**.
2. modifier vos fichiers sur votre machine pour faire votre BE par exemple
3. mettre à jour le dépôt **global** en copiant votre dépôt **local** dans le dépôt **global**

2 Première étape : connexion par clé SSH

2.1 En premier lieu

se connecter à <https://gitlab.ensimag.fr/> avec son login habituel

2.2 Générer une clé ssh

Sur votre machine (machine virtuelle Phelma, linux ou macos) dans une fenêtre *terminal* :

1. placez-vous sur votre répertoire home

```
[phelma@localhost ~]$ cd ~/
```

ou

```
[phelma@localhost ~]$ cd
```
2. générez une clé ssh en tapant


```
[phelma@localhost ~]$ ssh-keygen
```

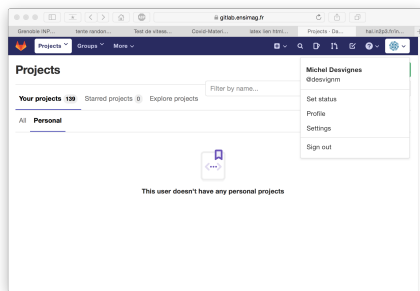
Attention : répondez par "Entrée" à toutes les questions
3. dans le répertoire `/.ssh/` vous devriez trouver 3 fichiers

```
[phelma@localhost ~]$ ls ~/.ssh/
```

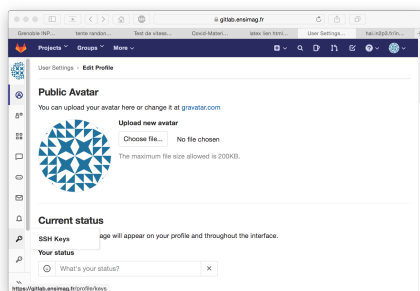
- le fichier `id_rsa` : c'est la clé privée qu'il ne faut jamais divulguer car on peut se faire passer pour vous.
- le fichier `id_rsa.pub` ; c'est la clé publique que vous pouvez divulguer, en particulier à la machine `gitlab.ensimag.fr` pour qu'elle puisse vous reconnaître. Il faudra la copier sur le serveur gitlab.
- le fichier `known_hosts` : c'est la liste des clés publiques des serveurs auxquels vous vous connectez en ssh. Ce fichier n'existe pas encore si vous n'avez jamais fait ce type de connexions.

2.3 Partager votre clé publique

1. Se connecter à <https://gitlab.ensimag.fr/> avec son login habituel si ce n'est pas fait.
2. Cliquer sur l'icône en haut à droite du bandeau. 
3. Sélectionner le menu *Settings* dans la fenêtre qui apparaît



4. Sélectionner le menu *SSH Keys*, 8^{ième} icône dans la barre de gauche la fenêtre prend l'allure suivante.

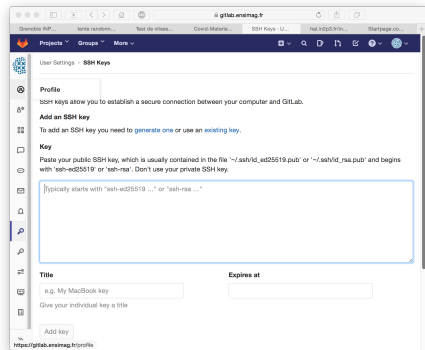


5. Sur votre machine, afficher la clé **publique** générée sur votre machine dans le fichier `id_rsa.pub`. Pour visualiser le contenu du fichier, utiliser l'éditeur `atom ~/.ssh/id_rsa.pub` ou la commande `cat ~/.ssh/id_rsa.pub`. La clé a la forme `ssh-rsa XXXXXXXX` ou `ssh-edXXXX`.



Sélectionner toute la clé y compris le mot-clé `ssh-rsa XXXXXXXX` ou `ssh-edXXXX` ainsi que le nom de la clé à la fin, à l'aide de la souris.

6. Copier la clé **publique** (sous Atom, menu Edit>Copy) dans le presse-papier
7. Coller la clé **publique** dans la fenêtre de `gitlab.ensimag.fr/`
8. Cliquer le bouton *Add Key* en bas à gauche.



Ca y est, votre clé publique est partagé entre le serveur *gitlab.ensimag.fr* et la machine sur laquelle vous travaillez actuellement.

Attention Chaque fois que vous voudrez travailler sur une nouvelle machine (hors phelma), vous devrez partager une nouvelle clé publique. Toutes les machines de phelma partagent la même clé publique.

2.4 Configurez votre environnement git

Pour chaque machine sur laquelle vous utiliserez git :

- `git config --global push.default simple`
- `git config --global user.name "Votre Nom"`¹
- `git config --global user.email you@example.com`
- `git config --global color.ui auto`
- `git config --global core.editor "atom --wait"`²

3 Création du dépôt local initial pour un be

Pour chaque nouveau be, vous allez devoir créer un nouveau dépôt local sur vos machines. Ce dépôt contiendra le sujet de be en version pdf, les fichiers de données si besoin. A chaque nouveau be, il faudra donc cloner le dépôt correspondant au be.

Il y a un dépôt par be, et son nom est `VOTRELOGINbe1`³, `VOTRELOGINbe2`, etc...

1. Vérifier que vous avez bien positionné les clés SSH (voir Première étape ci dessus)
2. Se connecter à <https://gitlab.ensimag.fr/> avec son login habituel
3. Accéder à l'ensemble de vos projets en cliquant sur le menu projects/Your Projects
4. Aller sur le projet `Phelma1A_info/VOTRELOGINbe1`
5. Cliquez sur le bouton bleu "Clone" et copier l'url de "Clone with SSH" qui ressemble à `git@gitlab.ensimag.fr:Phelma1A_info/VOTRELOGINbe1.git`
6. Sur votre machine, dans un terminal placez vous sur le répertoire `tdinfo`, situé sous le répertoire Documents sur la machine virtuelle Phelma⁴

```
[phelma@localhost ~]$ cd ~/Documents/tdinfo
```
7. Clonez le dépôt en utilisant la commande `git clone`, complétée en collant l'url obtenue ci dessus (étape 6) : la commande ressemble à

```
[phelma@localhost ~]$ git clone git@gitlab.ensimag.fr:Phelma1A_info/VOTRELOGINbe1.git
```

1. Mettez bien votre patronyme, pas la chaîne de caractère "Votre Nom"

2. Ou bien l'éditeur de texte que vous préférez

3. `VOTRELOGIN` doit bien sûr être remplacé par l'identifiant utilisé pour vos connexions, mais vous aviez deviné

4. Créer ce répertoire s'il n'existe pas encore : `[phelma@localhost ~]$ cd ~/Documents; mkdir tdinfo; cd tdinfo`

Si un message comme "Message authenticity of gitlab can't be established. Trust it ? yes / no" ou comme "The authenticity of host gitlab.ensimag.fr (195.221.228.226) can't be established. RSA key fingerprint is xxxxxxxxxxxxxxxxxxxx. Are you sure you want to continue connecting (yes/no)? " apparaît, répondez **yes** : . Ce message doit apparaître à la première tentative de clone sur le serveur gitlab. Lorsque vous répondez **yes**, la clé publique du serveur est stockée dans le fichier `known_hosts` et cette question ne sera plus posée.

Si le serveur vous demande votre mot de passe, c'est que la clé SSH est mal copiée sur le serveur gitlab.ensimag.fr ou que vous n'avez pas répondu **yes** à la question sur authenticity of host gitlab.ensimag.fr. Recommencez la même commande et répondez **yes** cette fois, ou partagez correctement votre clé publique en recommençant l'étape de partage de la clé publique.

8. Le répertoire `VOTRELOGINbe1` doit avoir été créé et contenir les fichiers du premier BE doivent être présents. Vérifier en utilisant les commandes :

```
[phelma@localhost ~]$ ls
```

qui doit faire apparaître le répertoire `VOTRELOGINbe1`
et la commande

```
[phelma@localhost ~]$ ls -l VOTRELOGINbe1
```

qui montre les fichiers du répertoire `VOTRELOGINbe1`.

9. Pour chaque machine sur laquelle vous voudrez travailler, il faudra commencer par *cloner* le dépôt **global**. Sauf cas exceptionnel, vous ne devez pas cloner le même dépôt 2 fois sur la même machine.

4 Comment travailler et réaliser ses BE

Il faut bien comprendre le principe de fonctionnement : le dépôt **global** doit contenir la version à jour de vos fichiers. Quand on travaille à plusieurs programmeurs ou avec plusieurs machines, c'est ce dépôt qui contient la bonne version.

Quand on commence une nouvelle séance de travail, la version locale, sur notre machine, n'est peut-être pas la dernière. Par exemple, si vous avez travaillé à Phelma sur les machines des l'école, les fichiers sur votre machine personnelle ne sont pas à jour. Il faut donc commencer par les mettre à jour.

Vous pouvez ensuite travailler sur vos fichiers, les modifier, les créer, les supprimer et réaliser vos BE.

A la fin de la séance de travail, ce sont alors les fichiers du dépôt **global** qui ne sont plus à jour. Il faut, avant de quitter votre séance, les mettre à jour.

Il faut donc toujours respecter ces 3 étapes :

1. Mise à jour du dépôt local
2. Modification des fichiers sur votre machine
3. Mise à jour du dépôt global

4.1 Mise à jour du dépôt local

La commande `git` qui permet de récupérer la dernière version des fichiers du dépôt **global**, à faire chaque fois que vous commencez à travailler est :

1. `git pull`

4.2 Mise à jour du dépôt global

Les 3 commandes `git` qui permettent de mettre à jour le dépôt **global** avec la dernière version des fichiers du dépôt **local** sont :

1. `git add les_fichiers_modifies`
2. `git commit -m "message indiquant les principales modifications faites"`
3. `git push`

4.3 Quelques commandes git utiles

Parmi les nombreuses commandes de git, voici les plus utiles :

- `git rm` : supprimer un fichier du depot **local**
- `git mv` : renommer un fichier du depot **local**
- `git status` : connaitre l'état du depot **local** ; indique quels sont les fichiers modifiés, lesquels ne sont pas encore envoyés sur le depot **global**
- `git log` : visualise toutes les modifications faites et par qui sur le dépôt **global**

En cas de problèmes avec git, une FAQ (Frequently Asked Questions) dédiée aux problèmes sous git est disponible sur le site tdinfo.phelma.grenoble-inp.fr. Cherchez si votre problème n'est pas déjà répertorié dans cette liste

5 Accéder aux autres sujets de BE et aux fichiers fournis

Pour accéder par la suite aux nouveaux BE et fichiers fournis, il faudra alors créer un nouveau dépôt pour chaque nouveau BE sur chaque machine sur laquelle vous travaillez.