

B3 TP GIT

Exercice 1 : Authentification

Création Clefs SSH

```
{10:30}/opt ➔ mkdir gitb3
{10:31}/opt ➔ cd gitb3
{10:31}/opt/gitb3 ➔ ssh-keygen -f id_rsa
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa
Your public key has been saved in id_rsa.pub
The key fingerprint is:
SHA256:73btwglqimPlqUrd7ZNp4ly7L+XmCu6piiyGIDqjyDs root@CCLP05CG032792Z
The key's randmart image is:
+----[RSA 3072]-----+
      Home
      S
      0 . . . .0
      = . .00.0*.0 0
      XE o++=0o+ = .
      B*++*B=-X+. o.
+----[SHA256]-----+
{10:31}/opt/gitb3 ➔ ll
total 8.0K
-rw----- 1 root root 2.6K Oct 14 10:31 id_rsa
-rw-r--r-- 1 root root 574 Oct 14 10:31 id_rsa.pub
{10:31}/opt/gitb3 ➔ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC34nYrozPIQ3GYCbsqsSof5RGOPq7tKvud1Qqb3+voQS1WvLTUUAIgnY
W5aT7rOSTnIdi5xEjVR5qUUMIjGwO6DEkczLre+uSpZFkZ6KQfRnwxaPNeItHhYe6udFPqZ7Ajj/JsquALOepqw1cQv7a8
LsP9AwgQlz9/urblNFwgXPN/EU7OWav22T2cHTe126vV2fPFjjOUryfMFZinukt8afq+JnC+7+92nTZqmSGJwg0TzBxMB6
bYHW8c2j1S2W6XgGUGZErObhS9fj8LYVNTkZTGCiFsUeWI1wK70xQfviEVch6tGEguyChXPldjPMMh3umYC0Bcb1UqDb7
8+JwCC3qXUi0H95Ny9FIi/S0veZasx4mz+NbxKqS1SOB14R96njfLGiPxODiA3pZH7yIBhTrEcmqbF9swQLddUsh1ryDBg
qbjz+t/pOMxiWJUI/Kc8mEHUByW/OYF4iIXrh/YAkrHRTaP/wgu6JyczS8n5XKhiwJaBOEXmQmNgh7Pf8= root@CCLP05
CG032792Z
{10:31}/opt/gitb3 ➔
```

Actions
Environments
Secrets
Pages
Moderation settings

Deploy keys / Add new

Title

git\_b3

Key

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC34nYrozPIQ3GYCbsqsSof5RGOPq7tKvud1Qqb3+voQS1WvLTUUAIgnY W5aT7rOSTnIdi5xEjVR5qUUMIjGwO6DEkczLre+uSpZFkZ6KQfRnwxaPNeItHhYe6udFPqZ7Ajj/JsquALOepqw1cQv7a8 LsP9AwgQlz9/urblNFwgXPN/EU7OWav22T2cHTe126vV2fPFjjOUryfMFZinukt8afq+JnC+7+92nTZqmSGJwg0TzBxMB6 bYHW8c2j1S2W6XgGUGZErObhS9fj8LYVNTkZTGCiFsUeWI1wK70xQfviEVch6tGEguyChXPldjPMMh3umYC0Bcb1UqDb7 8+JwCC3qXUi0H95Ny9FIi/S0veZasx4mz+NbxKqS1SOB14R96njfLGiPxODiA3pZH7yIBhTrEcmqbF9swQLddUsh1ryDBg qbjz+t/pOMxiWJUI/Kc8mEHUByW/OYF4iIXrh/YAkrHRTaP/wgu6JyczS8n5XKhiwJaBOEXmQmNgh7Pf8= root@CCLP05 CG032792Z


☒ Allow write access

Can this key be used to push to this repository? Deploy keys alw

Add key

Explication : ssh-keygen permet de créer une paire de clef SSH (Privée et publique). Sans arguments les clefs seront par défaut créer dans le dossier personnel de l'utilisateur. On pousse la clef publique sur le service sur lequel on souhaite s'authentifier. Lors des connexions notre clef privée servira à nous authentifier grâce à la clef publique présente sur github.

Ajout Clef Publique sur Github



git\_b3

SHA256:73btwglqimPlqUrd7ZNp4ly7L+XmCu6piiyGIDqjyDs

Added on 14 Oct 2021 by @Allta

Never used — Read/write

Delete

Explication : La clef publique a bien été enregistré sur Github. On peut voir qu'elle n'a jamais été utilisé encore. Une fois la clef enregistré on ne peut pas modifier les permissions donc il faut être vigilant lors de la création.

Paramétrage de Git

```
{10:39}/opt/gitb3 ➔ git config --global user.name "Allta"
{10:39}/opt/gitb3 ➔ git config --global user.email "francois.delrue@ynov.com"
{10:39}/opt/gitb3 ➔
{10:39}/opt/gitb3 ➔
{10:39}/opt/gitb3 ➔ cat ~/.gitconfig
[user]
  name = Allta
  email = francois.delrue@ynov.com
{10:40}/opt/gitb3 ➔
```

Explication : Paramétrage de git. Le flag global signifie que ces options (name et email) seront utilisé pour tout les repos git présent sur ma machine. On peut retrouver ces infos dans le fichier : ~/.gitconfig. Des configurations sont possible uniquement pour un repo git il faut faire les modifications dans le fichier ~/myrepo/.git/gitconfig.

Exercice 2 : Création d'un Repo Github

Initialiser votre dépôt local

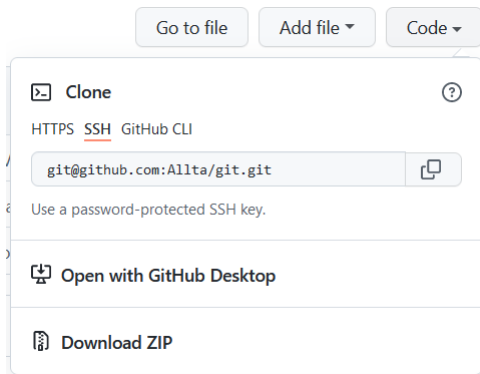
```
{10:44}/opt/gitb3 ➔ git init
Initialized empty Git repository in /opt/gitb3/.git/
{10:44}/opt/gitb3:master ✗ ➔
```

Explication : Initialisation d'un repo git. Il s'agit d'un dossier Linux standard qui est maintenant suivi par Git grâce au dossier .git. Ce dossier contient toutes les informations du depot local et distant. On voit que Git est sur la branche master. Il s'agit de la branch par défaut de git.

```
{10:51}/opt/gitb3:master ✗ ➔ git branch
{10:51}/opt/gitb3:master ✗ ➔ ll .git/branches
total 0
{10:51}/opt/gitb3:master ✗ ➔
```

Explication : Cependant il n'y a pas de remote paramétré encore donc pas de branch réellement configurée.

Faire pointer le dépôt local sur un dépôt distant (Remote)



Explication : Nous avons poussé la clef SSH donc nous allons utiliser l'URL **ssh** pour se connecter à notre remote.

```
{10:51}/opt/gitb3:master X ↵ git remote add origin git@github.com:Allta/git.git
```

Explication : Nous ajoutons le depot distant en tant que remote dans notre dossier git. Maintenant git sait sur quel repository pousser lors de nos commits.

#### Vérifier le dépôt distant

```
{10:51}/opt/gitb3:master X ↵ git remote add origin git@github.com:Allta/git.git
```

Explication : Nous pourrions utiliser le nom **origin** ou l'URL

#### Pull le repository

```
{11:01}/opt/gitb3:master X ↵ git pull origin main
warning: Pulling without specifying how to reconcile divergent branches is
discouraged. You can squelch this message by running one of the following
commands sometime before your next pull:

  git config pull.rebase false  # merge (the default strategy)
  git config pull.rebase true   # rebase
  git config pull.ff only       # fast-forward only

You can replace "git config" with "git config --global" to set a default
preference for all repositories. You can also pass --rebase, --no-rebase,
or --ff-only on the command line to override the configured default per
invocation.

remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 78 (delta 20), reused 3 (delta 1), pack-reused 0
Unpacking objects: 100% (78/78), 19.87 KiB | 190.00 KiB/s, done.
From github.com:Allta/git
 * branch            main       -> FETCH_HEAD
 * [new branch]      main       -> origin/main
{11:01}/opt/gitb3:master X ↵ ll
```

Explication : Pour puller un remote il faut spécifier le nom du remote ainsi que la branche. Cependant nous avons une disparité entre le remote et le local. La branche sur le remote est **main** (Pour Github main est la branche par défaut) et en local nous avons **master**. Lorsque nous allons pousser nous allons donc créer une nouvelle branche sur le remote. Pour pallier à cet effet nous aurions pu créer une nouvelle branche **main** avant de pull le repo. Ou lors de l'init nous aurions pu initialiser le repo avec la branche par défaut main :

### Exercice 3 : Modification du Projet CLI

#### Modifier un fichier déjà existant

Ajouter le port 2222 pour l'host Tyrell

```
modif
```

Set le niveau de log à INFO pour tout les hosts finissant en ell

```
mofig
```

#### Consulter le statut et les différences et Commiter vos modifications

```
commit_cycle
```

Explication : voici un cycle par default pour faire des modifs dans un projet git.

```
git_push
```

Explication : Git push permet de pousser nos commits locaux sur le remote en spécifiant la branche.

```
merge_request
```

Explication : Une merge request permet de fusionner 2 branches git après avoir revu les modifications. La merge request permet aussi de valider du code avant de le pousser en production.

#### Ajouter un fichier diff.txt et commencer à le tracker

Expliquer rapidement la différence entre git pull et git fetch

```
diff.txt
```

```
{11:08}/opt/gitb3:master X ↵ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    diff.txt
    id_rsa
    id_rsa.pub

nothing added to commit but untracked files present (use "git add" to track)
{11:08}/opt/gitb3:master X ↵ git add diff.txt
{11:08}/opt/gitb3:master X ↵ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   diff.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    id_rsa
    id_rsa.pub

{11:08}/opt/gitb3:master X ↵
```

*Explication :* Grâce à git add on peut ajouter spécifiquement un fichier au suivi par git. Ces fichiers seront maintenant considéré comme partie du projet et seront affecté par chaque action git.

#### Pousser vos modifications

```
{11:19}/opt/gitb3:main ✓ ↵ git checkout -b main
```

*Explication :* Git checkout permet de basculer d'une branche à une autre. Le travail sous branche permet de travailler parallèlement sans toucher à notre version en production

```
{11:18}/opt/gitb3:main ✓ ↵ git branch
* main
  master
```

*Explication :* Permet d'afficher les différentes branches de notre projet

```
{11:19}/opt/gitb3:main ✓ ↵ git pull origin main
```

*Explication :* Permet de tirer le code présent sur le remote d'une branche spécifique. Il faut prendre pour reflexe de tirer régulièrement le code de la branche de production lors de la création d'une nouvelle branche.

*Explication :* Permet de pousser du code sur notre remote. Il faut préciser la branche en upstream sur le repo local. Git nous l'indique de toutes façons.

#### Exercice 4 : Modification du projet GUI

##### Modifier le fichier config en se connectant sur l'interface Github

#### Exercice 5 : Branching et Merging

##### Pull le repo

*Explication :* La bonne pratique lors de developpement d'un projet est de pull la branche main/master pour être sur de travailler sur un repo à jour.

##### Assurer vous d'avoir un repo à jour

```
{11:23}/opt/gitb3:main ✓ ↵ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
{11:23}/opt/gitb3:main ✓ ↵
```

*Explication :* Normalement une fois qu'on pull un projet à jour notre status est propre.

##### Créer une nouvelle branche et basculer dessus

*Explication :* git checkout -b permet de créer et de basculer sur une branche directement, sinon nous aurions pu créer la branche avec **git branch**

##### Modifier la configuration SSH

##### Commiter et pousser la modification dans la nouvelle branche

*Explication :* **git diff** permet de voir les modifications sur les fichiers locaux dans une branche donnée.

##### Créer une Pull Request sur Github

*Explication :* Une pull request de Github est équivalent à une merge request sur Gitlab

##### Merger les 2 branches sur votre repo. (Avec les modifications de votre binome)



#### Continuous integration has not been set up

[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.



#### This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request



You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

*Explication:* Les pull requests permettent de déclencher des pipelines. Cependant ici ce n'est pas le cas



#### Pull request successfully merged and closed

You're all set—the `change_throne` branch can be safely deleted.

Delete branch

*Explication:* Lors d'une pull request nous pouvons supprimer la branche qui a servi à pousser les mods pour éviter d'avoir trop de branches sur des projets volumineux.



**Allta** deleted the `change_throne` branch 32 seconds ago

Restore branch