

GITHUB

Définition :

GitHub est une plateforme en ligne qui permet de stocker, gérer et collaborer sur des codes sources. En plus de servir de dépôt pour les projets de programmation, GitHub offre des outils pour suivre les modifications de code, gérer les versions, et collaborer avec d'autres développeurs. Elle est construite autour de **Git**, un système de contrôle de version qui permet de garder une trace des modifications de code et de gérer différentes versions d'un projet.

Installation :

Sur Windows :

1. Téléchargez l'installateur Git pour Windows depuis le site officiel : [Git for Windows](#).
2. Exécutez le fichier d'installation téléchargé et suivez les instructions.
3. Après l'installation, vous pouvez vérifier en ouvrant **Git Bash** (installé par défaut) ou en utilisant la commande :

git --version

Sur Ubuntu ou une autre distribution Lin

1-Mettez d'abord à jour la liste des paquets :

sudo apt update

2- Installez Git :

sudo apt install git

3-Vérifiez l'installation :

git --version

Sur macOS :

1. Installez Git via Homebrew (si Homebrew est installé) :

brew install git

2. Vérifiez l'installation :

git --version

Configuration initiale de Git

Une fois Git installé, il est recommandé de configurer votre nom d'utilisateur et votre adresse e-mail (ils apparaîtront dans chaque commit) :

```
git config --global user.name "VotreNom"
```

```
git config --global user.email "votreemail@example.com"
```

Différences entre Git et GitHub

- **Git** est un outil de gestion de versions qu'on utilise localement sur une machine pour suivre les changements des codes.
- **GitHub** est une plateforme en ligne qui utilise Git pour le partage et la collaboration sur des projets, avec des fonctionnalités supplémentaires pour la gestion des équipes et des contributions.

Quelques commandes en console:

Commande	Explication
C:\wamp\www\test\crud> git init	Initialisation du GIT
C:\Depot-git>git init --bare	Pour dire que le répertoire Depot-git est un depot de données pas un répertoire de travail
C:\wamp\www\test\crud>git remote add origin C:\Depot-git	Prendre comme depot C:\Depot-git sous le nom origin à partir du répertoire de travail C:\wamp\www\test\crud
C:\wamp\www\test\crud> git remote -v	Afficher les différents dépôts
C:\wamp\www\test\crud>git remote rename origin depot	Change le remote origin en depot
git remote add voyages https://github.com/votrepseudo/voyages.git	relie notre repository au repository distant
C:\wamp\www\test\crud>git remote rm origin_depot	Supprimer le remote origin_depot
C:\wamp\www\test\crud> git status	Afficher les statuts des fichiers : ils se mettent en rouge au départ
C:\wamp\www\test\crud> git add * ou C:\wamp\www\test\crud> git add nom_fichier	Rajouter les fichiers modifiés à l'index du serveur git. A cet effet les noms des fichiers se mettent en vert
C:\wamp\www\test\crud>git commit -m "Essai sur git"	Préparation de l'envoi des fichiers en vert avec message.
C:\wamp\www\test\crud>git commit -a -m "Essai sur git"	Combinaison de add et commit
git commit --amend -m "Votre nouveau message"	Changer le message
git revert SHADuCommit	Créer un nouveau commit à l'inverse du précédent
git reset --hard	Je veux annuler tous les changements que je n'ai pas encore commités
git clone https://github.com/JPBOTO/GIT_DEMO.git	Clonage en https de ma machine à mon GitHub à distance
git push origin master	Copie de master (moi) vers origine (ce qui est en clone)
git branch nouvelle-branche	Créer une nouvelle branche
git checkout nouvelle-branche	Pour se placer à la nouvelle branche Si la branche n'existe pas il le créer avant de s'y placer.
git branch	Pour afficher les différentes branches avec celle en cours marquée par un *
git merge brancheB	Fusionne le contenu de la branche brancheB à ma branche en cours que j'aurai du avoir par la commande "git checkout brancheA"
git blame nomdufichier.extension	Voir tous ce qui on modifier le fichier nomdufichier.extension A cet effet on a:

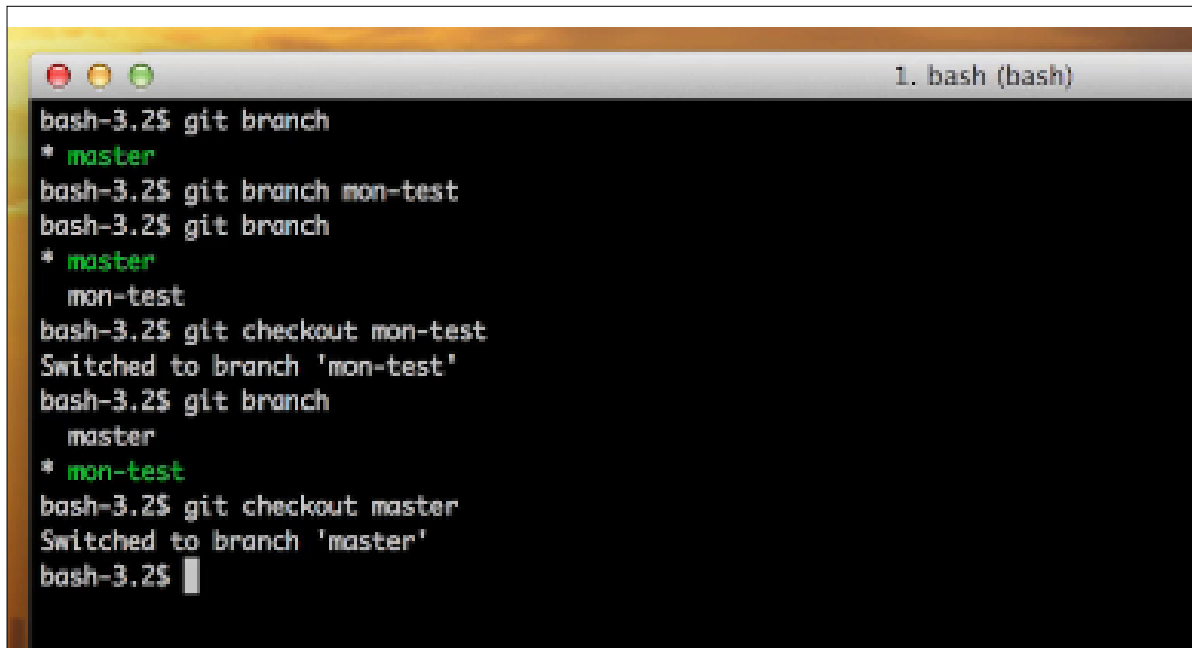
	^05b1233 (Marc Gauthier 2014-08-08 00:31:02 1) # Une liste
git show 05b1233	05b1233 est l'identifiant du commit. Cette commande affiche ce qui se passe dans ce commit
git stash	Mettre de côté vos modifications en cours avec la commande
git stash pop	Vous pouvez vous rendre dans la branche/le fichier que vous devez traiter à l'instant, finir et committer vos modifs. Une fois que vous avez réglé cette urgence, revenez sur la branche sur laquelle vous étiez en train de travailler, et récupérez les modifications que vous aviez mises de côté avec cette commande
git stash apply	Si vous voulez garder les modifications dans votre stash, vous pouvez utiliser apply à la place de pop
Fichier .gitignore	Mettre dans le fichier .gitignore tous les noms de fichier qu'on veut pas commiter comme parameters.yml, motdepasse.txt, ...
C:\wamp\www\test\crud> git push origin/master	
Git remote add origin https://github.com/JPBOTO/test.git	Envoie mon travail à mon git à distance
C:\wamp\www\test\crud>git branch git_avt	Créer une autre branche de nom git_avt dans mon espace de travail
C:\wamp\www\test\crud> git checkout git_avt Ou C:\wamp\www\test\crud> git checkout git shaDuCommit Ou C:\wamp\www\test\crud> git checkout git master	Aller à la branche git_avt Aller au cript correspondant à la commit Aller au branchement principal.(version la plus recente)
Git log	Afficher l'historique de commit

```
C:\wamp\www\TEST\CRUD>git remote add origin https://github.com/JPBOTO/TEST.git
```

```
C:\wamp\www\TEST\CRUD>git push -u origin master
Counting objects: 12, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (11/11), done.
```

```
Writing objects: 100% (12/12), 95.57 KiB | 0 bytes/s, done.
Total 12 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/JPBOTO/TEST.git
* [new branch]    master -> master
Branch master set up to track remote branch master from origin.
```

Voir : <http://www.copier-coller.com/installer-git-sur-windows/>

A screenshot of a terminal window titled "1. bash (bash)". The terminal shows a series of git commands and their outputs. The user starts by running "git branch", which shows "* master". Then they run "git branch mon-test", followed by another "git branch" command showing both "master" and "mon-test". Next, they run "git checkout mon-test", which switches to the "mon-test" branch. Then they run "git branch" again, showing "master" and "* mon-test". Finally, they run "git checkout master", switching back to the "master" branch. The terminal ends with "bash-3.2\$".

```
bash-3.2$ git branch
* master
bash-3.2$ git branch mon-test
bash-3.2$ git branch
* master
  mon-test
bash-3.2$ git checkout mon-test
Switched to branch 'mon-test'
bash-3.2$ git branch
  master
* mon-test
bash-3.2$ git checkout master
Switched to branch 'master'
bash-3.2$
```

Créer un dépôt qui servira de serveur

Si vous souhaitez mettre en place un serveur de rencontre pour votre projet, il suffit d'y faire un `git clone` ou un `git init` avec l'option `--bare`. Cela aura pour effet de créer un dépôt qui contiendra uniquement le dossier `.git` représentant l'historique des changements (ce qui est suffisant, car personne ne modifie les fichiers source directement sur le serveur).

`git init --bare` // À exécuter sur le serveur.

Pour se connecter au serveur, la meilleure méthode consiste à utiliser SSH. Ainsi, si vous voulez cloner le dépôt du serveur sur votre ordinateur, vous pouvez écrire quelque chose comme :

```
git clone ssh://utilisateur@monserveur.domaine.com/chemin/vers/le/depot/git //
à extraire sur la machine
```

De même, il faut configurer votre nom (ou pseudo) :

```
git config --global user.name "votre_pseudo"
```

Puis votre e-mail :

```
git config --global user.email moi@email.com
```

Installer Git sous Linux

Avec un gestionnaire de paquets, c'est très simple :

```
sudo apt-get install git-core gitk
```

Cela installe 2 paquets :

- git-core : c'est git, tout simplement. C'est le seul paquet vraiment indispensable ;
- gitk : une interface graphique qui aide à mieux visualiser les logs. Facultatif.

Il est recommandé de se créer une paire de clés pour se connecter au serveur de rencontre lorsque l'accès à Git se fait via SSH. Un tutoriel du SdZ explique comment créer des clés :

[http://www.siteduzero.com/tutoriel-3-7 \[...\] tml#ss part 5](http://www.siteduzero.com/tutoriel-3-7 [...] tml#ss_part_5).

Installer Git sous Windows

Pour utiliser Git sous Windows, il faut installer msysgit : <http://msysgit.github.io>.

Cela installe msys (un système d'émulation des commandes Unix sous Windows) et Git simultanément.