

Aucun cours ne remplace la documentation !

Doc Git : <https://git-scm.com/doc>

Doc Github : <https://docs.github.com>



I) Installation de votre environnement de travail Git et Github




- 1) Créez un compte **Github**¹ sur www.github.com
- 2) Téléchargez **Git**² sur <https://git-scm.com/downloads> et installez-le sur votre machine.
- 3) Configurez Git en ligne de commande :
`git config --global user.name "mon_pseudo_github"`
`git config --global user.email mon_email`
- 4) Vérifiez que vos paramètres de configuration ne comportent pas d'erreur : `git config --list`

¹ **Git** : logiciel de gestion de version installé sur votre machine qui vous permet de gérer vos dépôts locaux.

² **Git Hub** : plateforme web qui permet de gérer des dépôts distants afin de sauvegarder vos dépôts locaux et participer à des projets collaboratifs.

II) Création de votre premier repository depuis Github

En ligne de commande, placez-vous dans le dossier contenant les fichiers que vous souhaitez versionner.

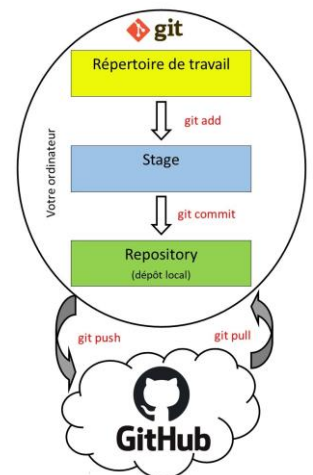
- 1) Sur votre Github, créez un nouveau repository .
- 2) Nommez-le, entrez une description, cochez la case "Add README.md" et cliquez sur .
- 3) Dans la page qui s'ouvre, cliquez sur  et copiez l'url (<https://github.com/monPseudo/monRepo.git>).

En ligne de commande, placez-vous dans le dossier où vous souhaitez placer votre dossier de travail.

- 4) Clonez votre dépôt distant sur votre machine : `git clone https://github.com/monPseudo/monRepo.git`
- 5) Vérifiez dans votre dossier qu'un nouveau dossier portant le nom de votre repository est présent.

III) Ajout de vos premiers changements à votre repository et envoi sur Github

- 1) En ligne de commande, placez-vous sur le nouveau dossier créé.
- 2) Modifiez le fichier README.md pour indiquer :
 - La description du projet.
 - La configuration requise.
 - Le guide d'installation.
 - Toutes autres informations utiles.
- 3) Ajoutez le fichier README.md au stage : `git add README.md`
- 4) Enregistrez ce changement dans votre dépôt local : `git commit -m "modify README.md"`
- 5) Poussez votre commit vers votre dépôt distant : `git push origin main`
- 6) Vérifiez sur Github que votre changement a bien été pris en compte.



Vous pouvez ainsi créer, supprimer et modifier autant de fichiers que vous le souhaitez.

La démarche pour les sauvegarder sera toujours la même :



- Ajoutez au stage toutes les modifications : `git add` . (ici le `.` indique qu'il faut stager tous les fichiers)
- Sauvegardez sur votre dépôt local : `git commit -m "message de commit"`
- Sauvegardez sur votre dépôt distant : `git push origin main` (généralement on push plusieurs commits en même temps)

Pour vérifier le statut de votre dépôt local : `git status`


Pour lister tous vos commits : `git log`

Pour lister vos commits et vos actions : `git reflog`

IV) Création de votre premier repository depuis votre machine

- 1) En ligne de commande, placez-vous dans le dossier contenant les fichiers que vous souhaitez versionner.
- 2) Sur votre Github, créez un nouveau repository .
- 3) Nommez-le, entrez une description et cliquez sur  et suivez les instructions dans la page qui s'ouvre.
- 4) Initialisez un nouveau dépôt Git : **git init** (cette commande crée un dossier .git caché)
- 5) Ajoutez tous les fichiers de ce dossier au stage : **git add .**
- 6) Commitez les fichiers ajoutés au stage : **git commit -m "mon message de commit"**
- 7) Changez le nom de la branche principale : **git branch -M main**
- 8) Indiquez que votre dépôt local doit pointer vers votre dépôt distant :
git remote add origin https://github.com/monPseudo/monRepo.git
- 9) Poussez vos commits vers votre dépôt distant : **git push -u origin main**
- 10) Vérifiez sur Gitub que vos fichiers sont sur votre repository.

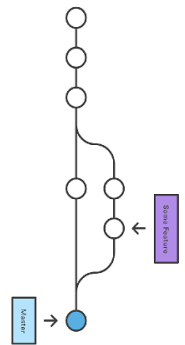
V) Récupération du projet d'un autre développeur

- 1) Rendez-vous sur le repository d'un autre développeur.
- 2) Cliquez sur  et téléchargez le zip du projet.
- 3) Dézippez ce fichier sur votre machine.
- 4) Suivez les étapes du IV "Créer votre premier repository depuis votre machine"

VI) Utilisation des branches pour une meilleure gestion de vos projets

Votre branche principale ne doit contenir que du code valide, testé et propre.

Par exemple, la version 3 de production d'une application se trouvera sur la branche principale, alors que la version 4 en développement se trouvera sur une branche secondaire. Cette branche secondaire ne sera fusionnée avec la branche principale, qu'une fois que l'application passera définitivement à sa version 4.



De même, pour ajouter une fonctionnalité à une application, on le fera dans une nouvelle branche :

- 1) Créez une nouvelle branche : **git branch nomBranche** (**nomBranche** doit expliciter la fonction développée)
- 2) Consultez les branches de votre dépôt local : **git branch** (la liste des branches s'affiche)
- 3) Placez-vous sur votre branche : **git checkout nomBranche**
- 4) Consultez à nouveau la liste des branches : **git branch** (notez l'* devant **nomBranche** indiquant que vous êtes dessus)
- 5) Effectuez des modifications et autant de commits que vous le souhaitez : **git push origin nomBranche**

Quand vous êtes sûr que votre fonctionnalité est terminée, fusionnez la branche créée à la branche principale :

- 6) Placez-vous sur la branche principale : **git checkout main** (après un **git branch**, l'* est sur main)
- 7) Fusionnez la branche secondaire sur la branche principale : **git merge nomBranche**
- 8) Vous pouvez maintenant supprimer si vous le souhaitez la branche secondaire : **git branch -d nomBranche**

VII) Quelques commandes utiles

git pull origin main permet de récupérer les modifications d'un dépôt distant sur la branche principale de votre machine.

git reset --hard HEAD^ permet de supprimer le dernier commit (pas encore poussé sur Github).

git commit --amend -m "nouveau message" permet de changer le message du dernier commit et/ou d'ajouter au dernier commit des nouveaux fichiers ajoutés au stage.