



Introduction : Clé SSH et Git pour les débutants

Un guide pour sécuriser et simplifier votre flux de travail de développement

Qu'est-ce qu'une clé SSH ?

SSH, ou Secure Shell, est un protocole réseau cryptographique permettant de faire fonctionner des services réseau de manière sécurisée sur un réseau non sécurisé. Une clé SSH est une méthode d'authentification pour ce protocole.

- Une paire de clés : Une clé privée que vous gardez secrète sur votre ordinateur, et une clé publique que vous partagez.
- Authentification sécurisée : Remplace l'usage de mots de passe, offrant une méthode de connexion plus robuste et pratique.
- Usages courants : Accès à des serveurs distants, services Git (comme GitHub, GitLab), transferts de fichiers sécurisés.



Génération d'une clé SSH sous Windows (pas à pas)

Sous Windows 10 et 11, vous pouvez utiliser l'outil OpenSSH intégré directement dans PowerShell ou l'Invite de commandes.

- Ouvrez PowerShell : Cherchez "PowerShell" dans le menu Démarrer et ouvrez-le.
- Lancez la commande : Tapez ``ssh-keygen -t ed25519 -C "votre_email@example.com"`` et appuyez sur Entrée.
- Choisissez l'emplacement : Appuyez sur Entrée pour accepter l'emplacement par défaut (``C:\\Users\\VotreNom\\.ssh\\id_ed25519``).
- Ajoutez une passphrase (recommandé) : Saisissez une phrase de passe pour une sécurité accrue. C'est un mot de passe pour votre clé privée.
- Vérifiez la création : Deux fichiers sont créés dans votre dossier ``.ssh`` : ``id_ed25519`` (la clé privée) et ``id_ed25519.pub`` (la clé publique).

Génération d'une clé SSH sous Pop!_OS (pas à pas)

La procédure est quasi identique sur la plupart des distributions Linux, y compris Pop!_OS, qui dispose nativement des outils nécessaires.

- Ouvrez un terminal : Utilisez le raccourci ``Ctrl+Alt+T`` ou cherchez "Terminal" dans vos applications.
- Lancez la commande : Exécutez ``ssh-keygen -t ed25519 -C "votre_email@example.com"``.
- Validez l'emplacement : Appuyez sur Entrée pour enregistrer la clé dans le répertoire par défaut ``~/.ssh/id_ed25519``.
- Définissez une passphrase : Entrez une phrase de passe sécurisée lorsque cela vous est demandé.
- Confirmez la création : Vous pouvez lister les fichiers avec ``ls -al ~/.ssh`` pour voir vos nouvelles clés.

Génération d'une clé SSH sous macOS (pas à pas)

macOS, étant basé sur Unix, suit une procédure très similaire à Linux. Le client OpenSSH est inclus par défaut.

- Ouvrez le Terminal : Allez dans `Applications` > `Utilitaires` > `Terminal`.
- Exécutez la commande : Tapez `ssh-keygen -t ed25519 -C "votre_email@example.com"`.
- Acceptez l'emplacement par défaut : Validez le chemin proposé, qui est `~/.ssh/id_ed25519`.
- Ajoutez une passphrase sécurisée : C'est une étape cruciale pour protéger votre clé privée.
- Ajoutez la clé à l'agent SSH : Pour éviter de retaper votre passphrase à chaque fois, exécutez `ssh-add -K ~/.ssh/id_ed25519`.

Connexion à Git via SSH : Configuration

Une fois la clé générée, vous devez fournir votre clé publique à votre service Git (GitHub, GitLab, Bitbucket) pour qu'il reconnaisse votre machine.

1. Copier la clé publique

Copiez le contenu de votre fichier de clé publique (`id_ed25519.pub`) dans votre presse-papiers. Utilisez la commande appropriée :


- **macOS:** `pbcopy < ~/.ssh/id_ed25519.pub`
- **Linux:** `xclip -selection clipboard < ~/.ssh/id_ed25519.pub` (installez xclip si besoin)
- **Windows:** `cat ~/.ssh/id_ed25519.pub | clip`

2. Ajouter à GitHub/GitLab

Connectez-vous à votre compte, puis :

- Allez dans vos `Paramètres` (Settings).
- Trouvez la section `Clés SSH et GPG` (SSH and GPG keys).
- Cliquez sur `Nouvelle clé SSH` (New SSH key).
- Donnez un titre descriptif (ex: "Mon PC Pop!_OS") et collez votre clé dans le champ principal.

Tester sa connexion SSH à Git

A dark-themed terminal window with the command 'ssh -T git:github.com' entered in a light-colored monospace font.

```
ssh -T git:github.com
```

Après avoir ajouté votre clé, vérifiez que la connexion fonctionne correctement. Cette étape est cruciale pour confirmer que tout est bien configuré.

- Ouvrez votre terminal ou PowerShell.
- Lancez la commande de test. Pour GitHub, tapez : ``ssh -T git@github.com``.
- Premier contact : Il est possible qu'on vous demande de confirmer l'authenticité de l'hôte. Tapez ``yes`` et validez.
- Message de succès : Vous devriez voir un message de bienvenue incluant votre nom d'utilisateur, confirmant que l'authentification a réussi. Par exemple : ``Hi username! You've successfully authenticated...``

C'est quoi Git ? Les bases

Git est un système de contrôle de version distribué. Pensez-y comme un système de sauvegardes intelligentes pour votre code, qui vous permet aussi de collaborer facilement avec d'autres.

Dépôt (Repository)

Le dossier de votre projet contenant tous vos fichiers ainsi que l'historique complet de toutes les modifications (dans un sous-dossier caché .git).

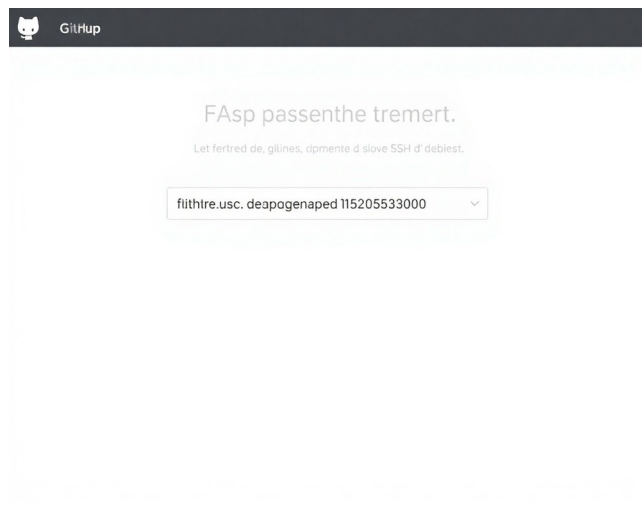
Commit

Un 'instantané' ou un 'point de sauvegarde' de votre projet à un moment donné. Chaque commit a un identifiant unique et un message descriptif.

Branche (Branch)

Une ligne de développement indépendante. Permet de travailler sur de nouvelles fonctionnalités sans affecter la version principale (généralement appelée 'main' ou 'master').

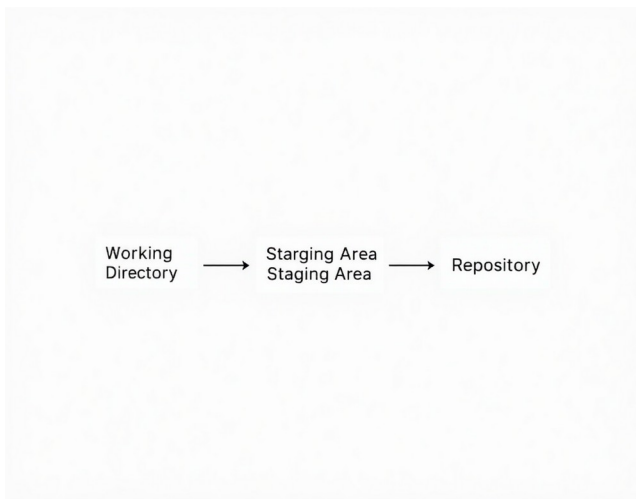
Cloner un dépôt via SSH



Maintenant que votre connexion SSH est fonctionnelle, vous pouvez l'utiliser pour cloner (télécharger) un dépôt. C'est plus sécurisé et évite de taper son mot de passe.

- Allez sur la page du dépôt (sur GitHub, GitLab...).
- Cliquez sur le bouton vert `Code`.
- Sélectionnez l'onglet `SSH`. L'URL doit commencer par `git@...` et non `https://...`.
- Copiez cette URL.
- Dans votre terminal, naviguez vers le dossier de votre choix et exécutez la commande : `git clone [URL copiée]`.

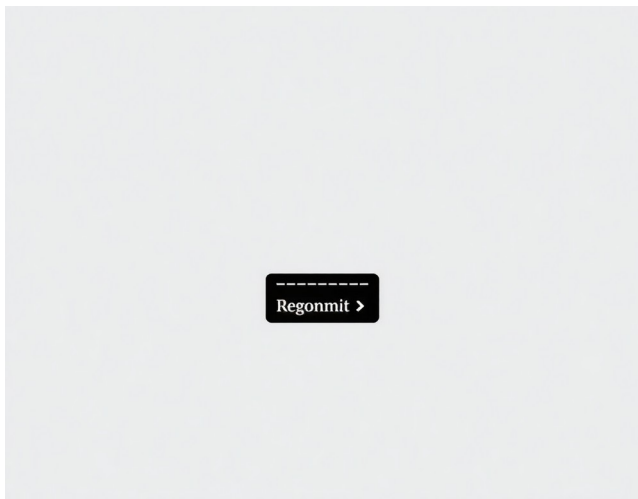
C'est quoi 'git add' ?



La commande ``git add`` sert à préparer les modifications que vous souhaitez inclure dans le prochain commit. Elle déplace les changements de votre répertoire de travail vers une zone de transit appelée 'staging area' (ou index).

- ``git add <nom_du_fichier>`` : Prépare les modifications d'un fichier spécifique.
- ``git add .`` : Prépare toutes les modifications (nouveaux fichiers, fichiers modifiés, fichiers supprimés) dans le répertoire courant et ses sous-répertoires.
- C'est une étape intermédiaire qui vous permet de choisir précisément ce qui fera partie de votre prochain 'point de sauvegarde' (commit).

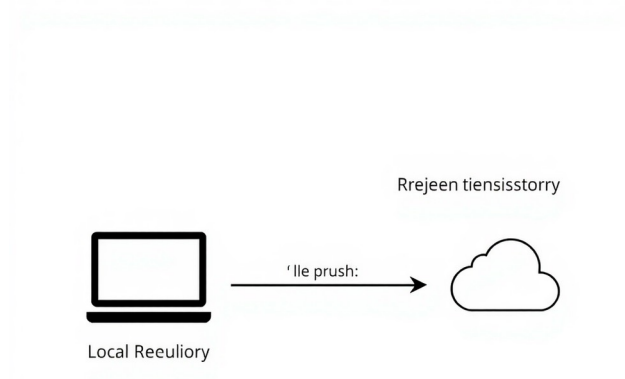
C'est quoi 'git commit' ?



La commande ``git commit`` enregistre de façon permanente les modifications qui se trouvent dans la 'staging area' dans l'historique de votre dépôt local. C'est l'acte de créer un point de sauvegarde.

- La commande est ``git commit -m "Votre message descriptif ici"``.
- Le ``-m`` permet de passer un message directement. Sans cela, Git ouvrira un éditeur de texte.
- Un bon message de commit est crucial : il doit expliquer le **pourquoi** du changement, pas seulement le **quoi**.
- Chaque commit est un maillon de la chaîne de l'historique de votre projet.

C'est quoi 'git push' ?



Après avoir créé un ou plusieurs commits sur votre machine locale, `git push` est la commande qui envoie ces commits vers le dépôt distant (par exemple, sur GitHub). C'est ainsi que vous partagez votre travail.

- La commande la plus courante est `git push origin main`, où `origin` est le nom par défaut de votre dépôt distant et `main` est le nom de la branche.
- C'est à ce moment que votre clé SSH entre en jeu pour vous authentifier de manière sécurisée auprès du serveur distant.
- Sans `git push`, vos commits restent uniquement sur votre ordinateur et ne sont pas visibles par vos collaborateurs.

Conclusion et prochaines étapes

Félicitations ! Vous maîtrisez désormais les bases pour utiliser Git avec SSH. C'est une compétence fondamentale qui vous servira tout au long de votre parcours de développeur.

Résumé

- Créer et configurer une clé SSH sur Windows, Linux et macOS.
- Lier sa clé SSH à un service Git comme GitHub.
- Comprendre le flux de base : cloner, ajouter, commiter, et pousser les modifications.

Prochaines Étapes

- ➔ Explorer les branches (``git branch``, ``git checkout``).
- ➔ Apprendre ``git pull`` pour récupérer les mises à jour.
- ➔ Se familiariser avec la résolution de conflits de fusion (``git merge``).
- ➔ Configurer un fichier ``.gitignore`` pour ignorer certains fichiers.