SUIVI ATELIER 1

Installation et configuration d'un GIT local

Apprendre les commandes de base

Rappel sur les 3 états d'un fichier

Vérification du statuts des fichiers

Ajout de fichiers (nouveau ou déjà suivi)

Ignorer des fichiers

Valider les changements

Suppression de fichiers

Déplacement de fichiers

Accès à l'historique

Assimiler les commandes d'annulation

1. Installation de Git :

Téléchargez et installez Git à partir du site officiel : https://git-scm.com/downloads

2. Configuration initiale :

Ouvrez une console/terminal et configurez votre nom d'utilisateur et votre adresse e-mail. Ces informations seront associées à vos commits.

```
git config --global user.name "Votre Nom"
git config --global user.email "votre@email.com"
```

3. Création d'un nouveau dépôt Git :

Créez un nouveau répertoire (ou utilisez un existant) pour votre projet et initialisez un dépôt Git.

```
mkdir MonProjet
cd MonProjet
git init
```

4. Les trois états d'un fichier :

Modifié (Modified) : Le fichier a été modifié depuis le dernier commit.

Indexé (Staged) : Les modifications ont été marquées pour être incluses dans le prochain commit.

Conservé (Committed) : Les modifications ont été sauvegardées dans la base de données Git.

5. Vérification du statut des fichiers :

Utilisez la commande git status pour voir l'état actuel de vos fichiers.

```
git status
```

6. Ajout de fichiers :

Pour ajouter un fichier modifié à la zone de staging :

```
git add nom_du_fichier
```

Pour ajouter tous les fichiers modifiés à la zone de staging :

git add .

7. Ignorer des fichiers :

Créez un fichier nommé .gitignore à la racine de votre dépôt et ajoutez les noms des fichiers ou dossiers que vous souhaitez ignorer. Par exemple, pour ignorer les fichiers texte, ajoutez "*.txt" dans le fichier .gitignore.

8. Commit des changements :

Une fois que vos fichiers sont en staging, vous pouvez les valider avec un commit.

```
git commit -m "Votre message de commit"
```

9. Historique des commits :

Utilisez la commande git log pour voir l'historique des commits.

```
git log
```

10. Validation des changements :

Pour valider les changements dans Git, utilisez la commande git commit après avoir ajouté les fichiers à la zone de staging.

```
git add nom_du_fichier

git commit -m "Votre message de commit"
```

11. Suppression de fichiers :

Pour supprimer un fichier du répertoire de travail et de la zone de staging, utilisez la commande git rm.

```
git rm nom_du_fichier

git commit -m "Supprimer le fichier"
```

12. Déplacement de fichiers :

Si vous déplacez ou renommez un fichier en dehors de Git, il peut ne pas reconnaître le changement. Pour déplacer/renommer un fichier tout en informant Git, utilisez git mv.

```
git mv ancien_nom nouveau_nom

git commit -m "Déplacer/Renommer le fichier"
```

13. Accès à l'historique :

Utilisez la commande git log pour voir l'historique des commits. Vous pouvez spécifier des options pour personnaliser la sortie.

git log

15. Commandes d'annulation :

Annuler les modifications locales non validées :

git checkout -- nom_du_fichier

Annuler les modifications dans la zone de staging (retourner au dernier commit) :

git reset HEAD nom_du_fichier

Annuler le dernier commit (attention : modifie l'historique) :

git reset --soft HEAD^

Annuler le dernier commit et les modifications locales (attention : modifie l'historique) :

git reset --hard HEAD^

16. Revenir à un commit spécifique :

Utilisez la commande git checkout pour revenir à un commit spécifique. Cela créera une branche détachée.

git checkout code_commit

17. Créer une nouvelle branche :

Pour travailler sur une nouvelle fonctionnalité ou corriger un bogue sans affecter la branche principale, utilisez les branches.

git branch nom_de_la_branche

git checkout nom_de_la_branche