Documentation des Commandes Usuelles de Git

Configuration initiale

Configurer votre nom d'utilisateur et votre adresse e-mail :

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

Initialiser un nouveau dépôt

Créer un nouveau répertoire, l'ouvrir et exécuter la commande suivante pour initialiser un nouveau dépôt Git :

```
git init
```

Cloner un dépôt existant

Pour cloner (copier) un dépôt existant :

```
git clone https://URL_DU_DEPOT.git
```

Vérifier l'état des fichiers

Pour voir quels fichiers sont modifiés ou en attente d'être ajoutés au commit :

```
git status
```

Ajouter des fichiers à l'index

Pour ajouter un fichier spécifique :

```
git add nom_du_fichier
```

Pour ajouter tous les fichiers modifiés :

```
git add .
```

Commiter les changements

Pour enregistrer les modifications dans le dépôt :

```
git commit -m "Message décrivant les modifications"
```

Pousser les changements

Pour envoyer vos commits vers le dépôt distant :

```
git push origin nom_de_la_branche
```

Tirer les changements

Pour mettre à jour votre dépôt local avec les dernières modifications du dépôt distant :

```
git pull
```

Branches

Créer une nouvelle branche :

```
git branch nom_de_la_nouvelle_branche
```

Changer de branche :

```
git checkout nom_de_la_branche
```

Combiner `git branch` et `git checkout` :

```
git checkout -b nom_de_la_nouvelle_branche
```

Fusionner des branches

Pour fusionner une branche dans votre branche actuelle :

```
git merge nom_de_la_branche
```

Voir l'historique des commits

Pour afficher l'historique des commits :

```
git log
```

Pour afficher un résumé en une ligne par commit :

```
git log --oneline
```

Annuler des modifications

Annuler les modifications dans un fichier avant de le stage (ajouter à l'index) :

```
git checkout -- nom_du_fichier
```

Retirer un fichier de l'index (mais garder les modifications) :

```
git reset nom_du_fichier
```

Revenir à un commit antérieur (danger de perte de données) :

```
git reset --hard commit_id
```

Aide

Pour obtenir de l'aide sur une commande Git :

```
git help nom_de_la_commande
```

Documentation Avancée des Commandes Usuelles de Git pour la Collaboration

Travail Collaboratif avec Git

Résoudre les conflits

Lorsque plusieurs personnes travaillent sur le même projet, il est commun que des conflits surgissent lors de fusions (`merge`) ou de tirages (`pull`).

Pour résoudre ces conflits :

1. Git marquera les fichiers en conflit. Ouvrez ces fichiers et cherchez les sections marquées par `<<<<<<<` , `=======` , et `>>>>>>>` .
2. Éditez les fichiers pour résoudre les conflits. Cela peut nécessiter une discussion avec les autres développeurs pour décider quelle modification garder.
3. Après avoir résolu les conflits, ajoutez les fichiers avec `git add` .
4. Complétez la fusion avec `git commit` . Git ouvrira un éditeur de texte pour vous permettre d'entrer un message de commit pour la fusion.

Pull Requests (Requêtes de Tirage)

Les pull requests ne sont pas une fonctionnalité directe de Git mais sont utilisées sur des plateformes comme GitHub, GitLab, et Bitbucket pour gérer les contributions.

1. Poussez votre branche vers le dépôt distant.
2. Sur la plateforme (par exemple, GitHub), cliquez sur le bouton "Create Pull Request" ou "New Pull Request".
3. Fournissez une description de vos changements et soumettez la pull request.

4. L'équipe peut alors passer en revue les modifications, discuter et demander des changements si nécessaire.
5. Une fois approuvée, la pull request peut être fusionnée dans la branche principale.

Tags

Les tags sont utilisés pour marquer des points spécifiques dans l'historique du projet, généralement pour marquer les versions des releases.

Pour créer un tag :

```
git tag -a v1.0 -m "Version 1.0"
```

Pour pousser les tags vers le dépôt distant :

```
git push origin --tags
```

Stashing

Si vous devez rapidement changer de contexte mais ne voulez pas commiter un travail inachevé, vous pouvez utiliser `git stash` pour mettre de côté vos modifications.

Pour mettre de côté vos modifications :

```
git stash
```

Pour réappliquer vos modifications mises de côté :

```
git stash pop
```