aquelito / **git-command.md**

Last active 8 days ago

GIT - Ligne de commande principale

git-command.md

Github

Rappel

Ne pas oublier la base : l'aide en ligne de commande. Il s'agit de la meilleur documentation.

```
git help config
git help push
git help pull
git help branch
```

Configuration

```
# Identity Name
git config --global user.name "aquelito"

# Identity Email
git config --global user.email "axel@aquelito.fr"

# Editor Tool
git config --global core.editor subl

# Diff Tool
git config --global merge.tool filemerge
```

Liste des globals

```
git config --list
```

Principales commandes

Status des fichiers

```
git status
```

Lister les branchs

```
git branch -a
```

* sur la branche courante.

Créer une branch

```
# Deux lignes: créer et basculer sur la nouvelle branch
git branch nom_de_ma_branch_nouvelle
git checkout nom_de_ma_branch_nouvelle
```

```
# Une seule ligne: créer et basculer
git checkout -b nom_de_ma_branch_nouvelle
```

Supprimer une branch

```
# Si la branch est local et n'est pas créée sur le repo distant
git branch -d nom_de_ma_branch_local

# Si la branch est présente sur le repo distant
git push origin --delete nom_de_ma_branch_distante
```

Changer de branch

```
git checkout nom_de_ma_branch
```

Premier commit

```
git add .
git commit -m "initial commit"
```

Commit suivant

```
git add chemin_vers_mon_fichier
git commit -m "message du commit"
```

Annuler le dernier commit et modifs

```
git reset --hard md5_commit
git push --force
```

Antidaté un commit.

```
git add .
GIT_AUTHOR_DATE="2015-12-12 08:32 +100" git commit -m "Commit antidaté"
```

Mettre à jour le dépôt local

```
git pull
```

Mettre à jour le dépôt local d'une branch spécifique

```
git pull origin MA_BRANCH
```

Envoyer ses commits vers le dépôt distant

```
git push
```

Envoyer ses commits vers le dépôt distant sur une branch spécifique

```
git push origin MA_BRANCH
```

Supprimer un fichier du répertoire de travail et de l'index

```
git rm nom_du_fichier
```

Supprimer un fichier de l'index

```
git rmg --cached nom_du_fichier
```

Diff

```
# Affiche la différence entre le contenu du dernier commit et celui du
# répertoire de travail. Cela correspond à ce qui serait commité par git commit -a.
git diff HEAD

# Affiche la différence entre le contenu pointé par A et celui pointé par B.
git diff A B

# Diff entre un dossier présent sur deux branches
git diff master..MA_BRANCH chemin/vers/mon_dossier
```

Log

```
# Classique
git log

# Affiche X derniers commits
git log -n X

# Affiche les commits concernant un dossier
git log --oneline -- chemin/vers/mon_dossier

# Affiche un ensemble de commits par date
git log --since=date --until=date

# Représentation de l'historique à partir de HEAD (commit / branch)
git log --oneline --graph --decorate

# Représentation de l'historique à partir d'un fichier (commit / branch)
git log --oneline --graph --decorate nom_du_fichier
```

Annuler commits (soft)

Seul le commit est retiré de Git ; vos fichiers, eux, restent modifiés. Vous pouvez alors à nouveau changer vos fichiers si besoin est et refaire un commit.

Annuler le dernier commit

```
git reset HEAD^
```

Pour indiquer à quel commit on souhaite revenir, il existe plusieurs notations :

- HEAD : dernier commit ;
- HEAD^ : avant-dernier commit ;
- HEAD^^ : avant-avant-dernier commit ;
- HEAD~2 : avant-avant-dernier commit (notation équivalente) ;
- d6d98923868578a7f38dea79833b56d0326fcba1 : indique un numéro de commit ;
- d6d9892 : indique un numéro de commit version courte.

Annuler commits (hard)

Si vous voulez annuler votre dernier commit et les changements effectués dans les fichiers, il faut faire un reset hard. *Cela annulera sans confirmation tout votre travail !*

Annuler les commits et perdre tous les changements

```
git reset --hard HEAD^
```

Annuler les modifications d'un fichier avant un commit

Si vous avez modifié plusieurs fichiers mais que vous n'avez pas encore envoyé le commit et que vous voulez restaurer un fichier tel qu'il était au dernier commit :

```
git checkout nom_du_fichier
```

Annuler/Supprimer un fichier avant un commit

Supposer que vous venez d'ajouter un fichier à Git avec `git add` et que vous vous apprêtez à le « commiter ». Cependant, vous vous rendez compte que ce fichier est une mauvaise idée et vous voulez annuler votre `git add` .

Il est possible de retirer un fichier qui avait été ajouté pour être « commité » en procédant comme suit :

```
git reset HEAD -- nom_du_fichier_a_supprimer
```

installer_un_projet.md

Installation

Initialiser un projet

```
cd chemin/vers/mon_dossier
echo "# MON_PROJET" >> README.md
git init
git add README.md
git commit -m "Initial commit"
git remote add origin ....git
git push -u origin master
```

Récuperer le repo sur Github

```
git clone ....git chemin/vers/mon_dossier
```

Mon repo est composé d'au moins deux branch.

develop : dédié au développement et résolution de bug. master : reflète le code en production. Personne ne doit travailler directement sur cette branch.

Pour récupérer votre branch develop

```
git branch -a
git checkout origin/develop
git checkout -b develop origin/develop
git branch
```

Developpement : Branch develop

Prod : Branch Master

```
# On se met sur la branche master
git checkout master
# On merge la branche develop
git merge develop
# Pour pousser les changements
git push origin master
# Penser à revenir sur develop
git checkout develop
```