



UTILISER GIT

TELECHARGER GIT

<https://git-scm.com/download/win>

INSTALLER GIT

Laissez l'installation standard si vous souhaitez utiliser uniquement la console GIT en cliquant simplement sur Next à chaque étapes.

Sinon pensez à mettre **“Use Windows default console window”** pour pouvoir utiliser l'invite de commandes Windows



OUTILS COMPLÉMENTAIRES

Interface graphique en NodeJs <https://github.com/FredrikNoren/ungit>

GIT C'EST QUOI ?

Git est un système de versionning, il vous permettra de sauvegarder les différentes étapes d'un projet il permet de suivre les historiques d'un développement et de gérer le "merging" (fusions des fichiers) afin de résoudre les conflits permettant ainsi de travailler efficacement à plusieurs sur un projet.

PASSER EN MODE CONSOLE

Si vous utilisez la console GIT

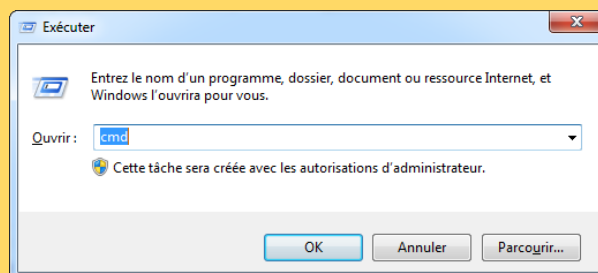
Démarrer > [Tous les programmes] > Git > Git CMD

Ou

Poste de travail > C:\Program Files\Git\git-cmd.exe

Si vous utilisez la console Windows

Pressez sur les touches **[windows] + [R]**, tapez "**cmd**" ou "**powershell**" et appuyez sur OK sur la fenêtre qui vient de s'ouvrir



Ou

Démarrer > [Tous les programmes] > Accessoires > Invites de commandes

INITIALISATION D'UN PROJET GIT en ligne de commande

Se placer dans le dossier souhaité (**cd C:/mon_dossier_git/**).

-- Créer au besoin un sous dossier (**mkdir "Nom du projet Git"**) et se placer dans le sous dossier (**cd "Nom du projet Git"**).--

Une fois dans le dossier faites :

```
git init
```

Capture :

OBTENIR DE L'AIDE SUR GIT

Toutes les commandes d'aides sont écritent de cette couleur.

[/ ? \ : git --help]

CONFIGURER SON GIT

[/ ? \ : git help config]

Déclarer son email

```
git config --global user.email "cyrhades76@gmail.com"
```

Déclarer son nom

```
git config --global user.name "Cyrhades"
```

Voir la config

```
git config --list
```

LES LIGNES DE COMMANDES INDISPENSABLE

Pour connaître l'état du GIT

```
git status
```

Pour voir les logs du GIT [/ ? \ : git help log]

```
git log
```

Pour ajouter un fichier [/ ? \ : git help add]

```
git add "nom_du_fichier"
```

Pour ajouter tous les fichiers dans le dossier

```
git add --all
```

Pour ajouter des types de fichier (exemple les fichiers ayant l'extension html)

```
git add "*.html"
```

Valider le commit [/ ? \ : git help commit]

```
git commit
```

Si vous ne mettez pas de commentaire une interface (un éditeur cf:capture) vous invite à le faire

Annuler un commit qui n'est pas encore push

Annule le commit, si on fait un git status on est revenu comme avant le git commit

```
git reset
```

Revenir à une ancienne version

On doit d'abord connaître les versions qui ont impacté ce que nous souhaitons modifier, pour se faire voici les commandes :

```
git log --oneline
```

Pour l'ensemble du projet

```
C:\GIT\projet_1>git log --oneline
3aa7731 MAJ 1.0
f31bed4 MAJ 2.0
3cc41a3 test
b930093 test
0c8481f Ajout des fichiers ignorés
4a35538 Modification index.php
cd37bbe premier commit
```

```
git log --oneline "nom_du_fichier"
```

Pour un seul fichier

```
C:\GIT\projet_1>git log --oneline index.php
3aa7731 MAJ 1.0
f31bed4 MAJ 2.0
4a35538 Modification index.php
cd37bbe premier commit
```

```
git log --oneline *.php
```

Pour un type de fichiers

```
C:\GIT\projet_1>git log --oneline *.php
3aa7731 MAJ 1.0
f31bed4 MAJ 2.0
3cc41a3 test
b930093 test
4a35538 Modification index.php
cd37bbe premier commit
```

Pour revenir à une version précédente du projet [/ ? \ : git help checkout]

(dans l'exemple ci dessous : premier commit)

```
git checkout cd37bbe
```

En restant sur la branche en cours

```
git checkout cd37bbe -b "nom_de_la_nouvelle_branche"
```

En créant une nouvelle branche

Revenir à la version précédente

```
git reset HEAD~1
```

POUSSER LES MODIFS SUR UN SERVEUR DISTANT

On ajoute l'origine

```
git remote add origin "adresse_distante/nom_du_projet.git"
```

Pousser la donnée

```
git push -u origin "nom_branche"
```

GÉRER LES BRANCHES

Créer une nouvelle branche

```
git branch "nom_de_la_nouvelle_branche"
```

Se placer sur une branche

```
git checkout "nom_de_la_branche"
```

Renommer une branche

```
git branch -M "nom_de_la_branche" "nouveau_nom"
```

("M" équivaut à "-m -f")

Supprimer une branche

```
git branch -D "nom_de_la_branche"
```

("D" équivaut à "-d -f")

TRAVAILLER AVEC L'HISTORIQUE

TODO : Je n'ai jamais continué ce Tuto, je m'y remettrais un jours promis ^^

Générer une clef SSH pour GitHub

TODO : Je n'ai jamais continué ce Tuto, je m'y remettrais un jours promis ^^