

# GIT



Pour ce cours vous pouvez retrouver les sources sur ce lien :  
<https://rebrand.ly/b7fnhz>



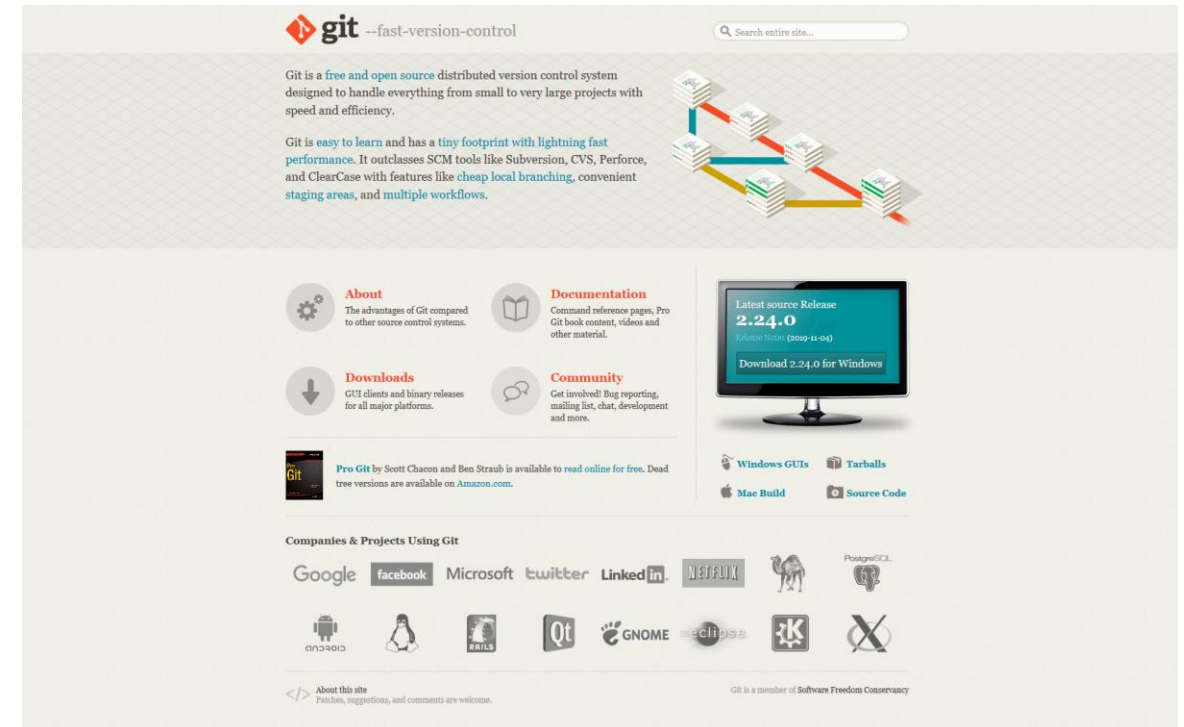
**Un logiciel de gestion de versions (ou VCS en anglais, pour version control system) est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus.**

**Il permet notamment de retrouver les différentes versions d'un lot de fichiers connexes.**

# Installation

**Pour utiliser GIT vous aurez besoin :**

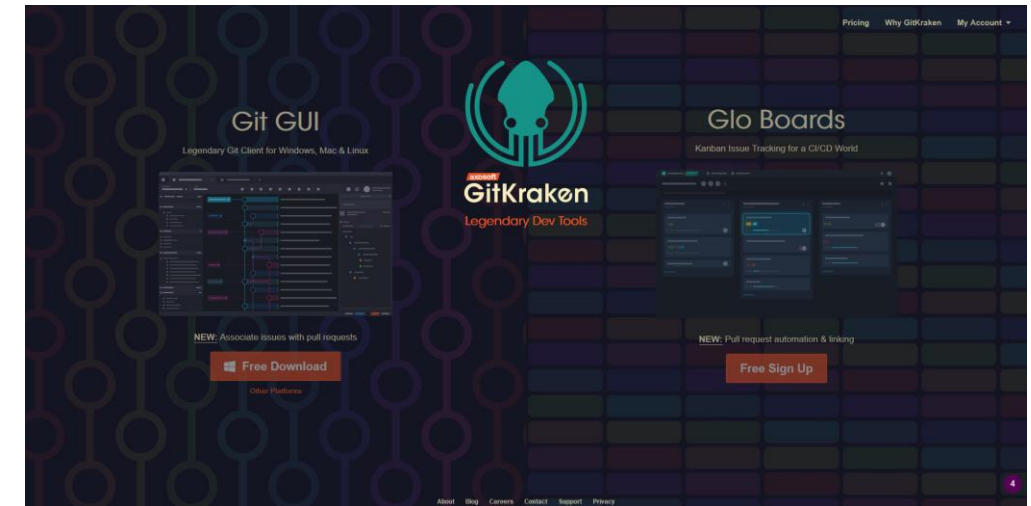
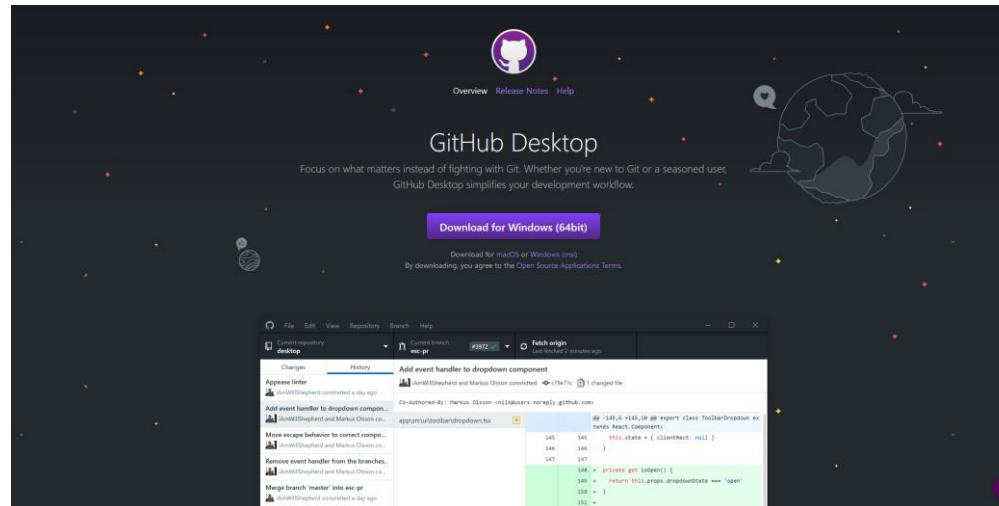
1. GIT installé en local (<https://git-scm.com>)
2. Un serveur de stockage (peut être en local ou vers une plateforme)



# Installation

Pour utiliser GIT vous aurez besoin :

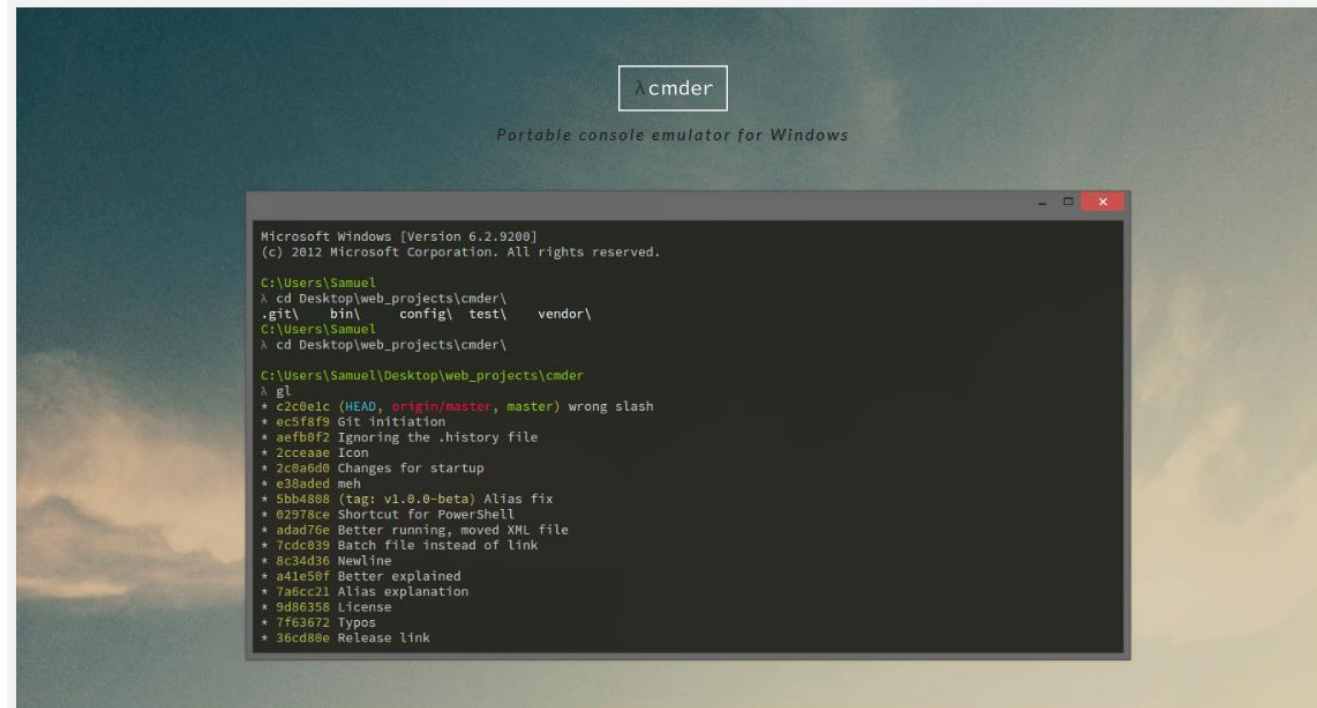
3. logiciels



# Installation

Pour utiliser GIT vous aurez besoin :

## 3. Logiciels (bonus)



```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Users\Samuel>
λ cd Desktop\web_projects\cmdr\
.git\ bin\ config\ test\ vendor\
C:\Users\Samuel>
λ cd Desktop\web_projects\cmdr\

C:\Users\Samuel\Desktop\web_projects\cmdr>
λ gl
* c2c0e1c (HEAD, origin/master, master) wrong slash
* ec5f8f9 Git initiation
* aefb0f2 Ignoring the .history file
* 2ccea9e Icon
* 2c0a6d0 Changes for startup
* e38aded meh
* 5bb4808 (tag: v1.0.0-beta) Alias fix
* 02978ce Shortcut for PowerShell
* adad76e Better running, moved XML file
* 7cdc939 Batch file instead of link
* 8c34d36 Newline
* a41e50f Better explained
* 7a0cc21 Alias explanation
* 9d06358 License
* 7f63672 Typos
* 36cd80e Release link
```

# Les commandes à connaître

**git init** : Initialise un nouveau dépôt Git.

**git config** : lance la procédure de configuration de Git.

**git help** : Affiche une liste de commande utile.

**git status** : Vérifie le statut de votre repository. Affiche les fichiers du repo, les modifications, la branche courante.

**git add** : Permet de prendre en compte les fichiers dans votre futur commit.

**git commit** : Permet de sauvegarder une image de l'instant T de votre code afin de l'envoyer plus tard vers le serveur. (git commit -m "Le message qui illustre vos modifs").

## Les commandes à connaître

**git branch** : Permet de créer une nouvelle « branche » afin de travailler sur une version parallèle (git branch branche\_de\_jerome).

**git checkout** : Permet de changer de branche (git checkout branche\_de\_jerome).

**git merge** : Permet de fusionner deux branches (git merge branche\_de\_jerome) va fusionner la branche\_de\_jerome avec la branche principale (master).

**git push** : Permet d'envoyer vos modifications vers le serveur.

**git pull** : Permet de récupérer la dernière version sur votre ordinateur d'une branche à jour.


**git clone** : Permet de récupérer un projet depuis un serveur et de le cloner sur son ordinateur.



# Processus et cycle de fonctionnement

Étape 1 : Créer un repo sur son serveur

Étape 2 : Créer le repo sur son ordinateur



```
echo "# cours-git" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin
https://github.com/jeromediaferia/cours-git.git
git push -u origin master
```

## Processus et cycle de fonctionnement

1

```
C:\Users\jerom  
λ cd C:\laragon\www
```

On se déplace dans notre dossier

2

```
C:\laragon\www  
λ mkdir cours-git
```

On crée un dossier pour notre projet

3

```
C:\laragon\www  
λ cd cours-git
```

On se déplace dans notre projet

4

```
C:\laragon\www\cours-git  
λ git init  
Initialized empty Git repository in C:/laragon/www/cours-git/.git/
```

On initialise le projet

## Processus et cycle de fonctionnement

5 `C:\laragon\www\cours-git (master -> origin)  
λ echo "# cours-git" >> README.md`

On crée un fichier readme.md

6 `C:\laragon\www\cours-git (master -> origin)  
λ git add README.md`

On ajoute notre fichier à notre GIT

7 `C:\laragon\www\cours-git (master -> origin)  
λ git commit -m "first commit"  
[master (root-commit) 97ab463] first commit  
1 file changed, 1 insertion(+)  
create mode 100644 README.md`

On commit (commente) notre modification

## Processus et cycle de fonctionnement

8

```
C:\laragon\www\cours-git (master -> origin)  
λ git remote add origin https://github.com/jeromediaferia/cours-git.git
```

On connecte le projet du serveur avec celui de notre ordinateur

9

```
C:\laragon\www\cours-git (master -> origin)  
λ git push -u origin master  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 240 bytes | 48.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0)  
To https://github.com/jeromediaferia/cours-git.git  
* [new branch]      master -> master  
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

On envoie nos fichiers vers le serveur

# Exercices

1

1. Créer un document : index.html
2. L'ajouter à votre repo
3. Commenter votre modification
4. Pousser la modification

## **Astuces**

- ✓ Vous pouvez utiliser « git add \* » pour ajouter tout vos fichiers modifiés

# Exercices

## 2

1. Créer une nouvelle branche « test »
2. Changer de branche
3. Modifier le fichier index.html
4. L'ajouter à votre repo
5. Commenter votre modification
6. Ajouter votre nouvelle branche vers votre serveur
7. Pousser la modification sur la branche « test »
8. Revenir sur la branche « master »
9. Fusionner la branche « test » dans la branche master
10. Pousser les modifications sur « master »

## 2.6

```
C:\laragon\www\cours-git (test -> origin)  
λ git remote add test https://github.com/jeromediaferia/cours-git
```

## 2.8

```
C:\laragon\www\cours-git (master -> origin)  
λ git merge test  
Updating abf3cc1..d296810  
Fast-forward  
 index.html | 3 ++-  
 1 file changed, 2 insertions(+), 1 deletion(-)
```

# Exercices

2

1. Créer une nouvelle branche « test »
2. Changer de branche
3. Modifier le fichier index.html
4. L'ajouter à votre repo
5. Commenter votre modification
6. Ajouter votre nouvelle branche vers votre serveur
7. Pousser la modification sur la branche « test »
8. Revenir sur la branche « master »
9. Fusionner la branche « test » dans la branche master
10. Pousser les modifications sur « master »

2.6

```
C:\laragon\www\cours-git (test -> origin)  
λ git remote add test https://github.com/jeromediaferia/cours-git
```

2.8

```
C:\laragon\www\cours-git (master -> origin)  
λ git merge test  
Updating abf3cc1..d296810  
Fast-forward  
 index.html | 3 ++-  
 1 file changed, 2 insertions(+), 1 deletion(-)
```

# Exercices

3

1. Revenir sur la branche « test »
2. Modifier la ligne 1 du fichier index.html
3. Ajouter la modification « git add »
4. « commit » le document
5. « push » le document
6. Changer de branche et revenir sur master
7. Fusionner la branche « test » dans la branche master
8. Résoudre le conflit
9. Pousser les modifications sur « master »



# Exercices

4

1. Revenir sur la branche « test »
2. Modifier de nouveau la ligne 1 du fichier index.html
3. Ajouter la modification « git add »
4. « commit » le document
5. Changer de branche et revenir sur master
6. Modifier la même ligne
7. Ajouter la modification
8. « commit » le document
9. Fusionner la branche « test » dans la branche master
10. Git annonce un conflit
11. Résoudre le conflit

4.10

```
C:\laragon\www\cours-git (master -> origin)
λ git merge test
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

C:\laragon\www\cours-git (master -> origin)
λ |
```

## Exercices

### 4.10

```
C:\laragon\www\cours-git (master -> origin)
λ git merge test
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

C:\laragon\www\cours-git (master -> origin)
λ |
```

Index.html de la branche master

### 4.11

```
1 <<<<<< HEAD
2 Modification 1.1.1
3 =====
4 Modification 1.1 conflit dans le test
5 >>>>>> test
6 Modification 2|
```

Index.html de la branche test

Dans le point 4.10, git remonte un conflit car le même fichier a été modifié sur deux branches. Quand on lui demande de fusionner, il nous le signale et nous dit qu'il faut régler le conflit avant de continuer. Dans notre fichier index.html, git nous montre avec les <<<< >>>> la zone en conflit. Il faut donc modifier notre fichier avec ce que l'on veut garder.

## Exercices

### 4.10

```
C:\laragon\www\cours-git (master -> origin)
λ git merge test
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

C:\laragon\www\cours-git (master -> origin)
λ |
```

Index.html de la branche master

### 4.11

```
1 <<<<<< HEAD
2 Modification 1.1.1
3 =====
4 Modification 1.1 conflit dans le test
5 >>>>>> test
6 Modification 2|
```

Index.html de la branche test

Dans le point 4.10, git remonte un conflit car le même fichier a été modifié sur deux branches. Quand on lui demande de fusionner, il nous le signale et nous dit qu'il faut régler le conflit avant de continuer. Dans notre fichier index.html, git nous montre avec les <<<< >>>> la zone en conflit. Il faut donc modifier notre fichier avec ce que l'on veut garder.

# Exercices

Pendant le conflit

```
1 <<<<<< HEAD
2 Modification 1.1.1
3 =====
4 Modification 1.1 conflit dans le test
5 >>>>>> test
6 Modification 2|
```

Après le conflit

```
1 Modification 1.1.1 + conflit dans le test
2 Modification 2|
```

12. Il ne reste plus qu'à ajouter les modifications puis de commit et enfin de push.