Bachelor CDWFS
Versionning
2025-2026

GIT

Commandes avancées

Status des fichiers

Working directory	Stage	Repository	Statut
Nouveau fichier			Untracked
git add .			
Nouveau fichier	Nouveau fichier		Staged
Update du fichier			
Fichier modifié	Nouveau fichier		Modified (& Staged)
git add .			
Fichier modifié	Fichier modifié		Staged
git commit			
Fichier modifié	Fichier modifié	Fichier modifié	Unmodiifed

Reproduire l'exemple et constater l'évolution du statut et le code couleur

Code douleur

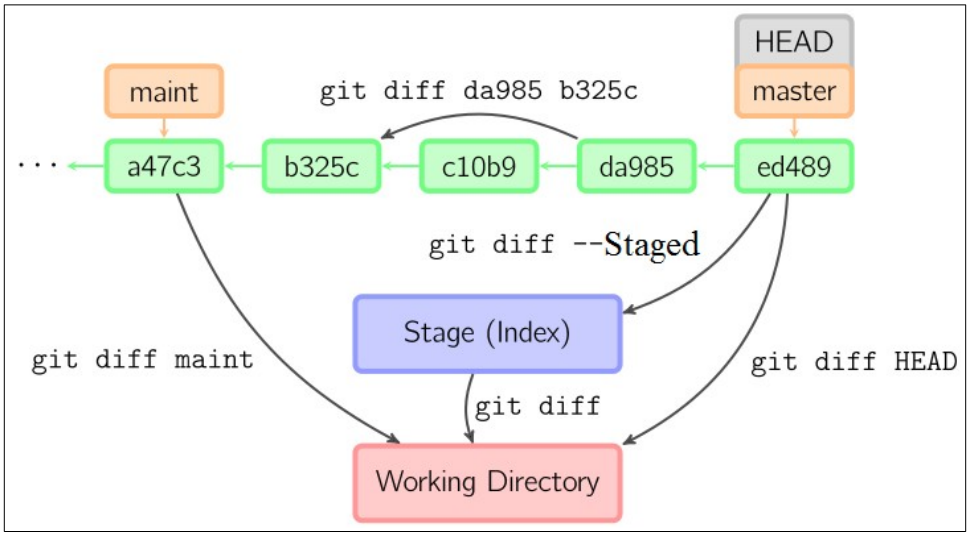
Statut	Couleur
Untracked	rouge
Staged	Vert
Modified (& Staged)	Rouge & vert

Commandes avancées

git diff

Cette commande permet de voir toutes les différences entre le répertoire de travail et le dépôt.

Syntaxes & usages



Zone « Répertoire de travail »

Restaurer

Possibilité de restaurer depuis un commit :

https://git-scm.com/docs/git-restore

Avec la commande restore

Depuis l'index :

Cette commande récupère la version actuelle du fichier **dans l'index** et la place dans le répertoire de travail :

```
git restore <file>
```

Depuis le dépôt :

Ce commande récupère la version actuelle du fichier **dans le dépôt git** et la place dans le répertoire de travail.

```
git restore --source=HEAD <fichier>
```

Avec la commande checkout

Effet identique à la commande précédente

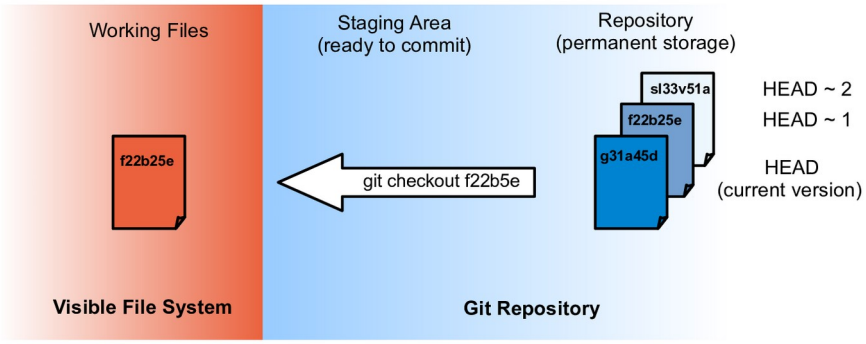
```
git checkout HEAD – <fichier>
```

Se déplacer dans l'historique

Avec la commande checkout

```
git checkout <hash>
```

Cette commande permet de visualiser l'état de l'application à un commit donné.
Pour lancer cette commande, le répertoire de travail doit être « **iso** » avec le dépôt git.



L'état « Tête détaché »

Cette commande déplace la tête HEAD de git, sur le commit spécifié. Le depôt git est dans un état « Tête détaché » **(c'est à dire qu'il pointe sur un commit, et non sur une branche).**

Pour sortir de l'état tête détaché, il faut déplacer HEAD sur une branche.
Exemple :

```
git checkout main
```

Zone « Index »

Lister les fichiers indexés

Avec la commande ls-files

Cette commande permet de lister les fichiers placés dans l'index

```
git ls-files <fichier>
```

Retirer des fichiers de l'index

Avec la commande restore

Cette commande permet d'annuler l'ajout d'un fichier dans l'index

```
git restore --staged <fichier>
```

Avec la commande reset

Cette commande récupère la version courante du fichier dans le dépôt git et la place dans la zone d'index.

```
git reset HEAD <file>
```

Avec la commande rm

Cette commande permet de retirer des fichiers de l'index

```
git rm --cached vendor
```

Zone « Dépôt »

Annuler un commit

Pour annuler le dernier commit :

```
git revert HEAD
```

Pour annuler un commit spécifique :

```
git revert 5b445ea
```

git revert créer un nouveau commit pour annuler un commit précédent
git revert annule le dernier commit et modifie l'index et le répertoire de travail en conséquence.

Zone « Remise »

Remiser des fichiers

Pour mettre de côté les modifications en cours dans le répertoire de travail :

```
git stash
```

Pour les récupérer :

```
git stash pop
```

Les dossiers

Les dossiers sont-ils « suivis » ?

A savoir :
Un dossier vide ne peut pas être indexé.

Pour être indexé, un dossier doit indexer au moins un fichier.

A. Romero

Exercice

Créer un dossier vide et la commande git status : constater que le dossier n'apparaît pas

A savoir :
La commande git status n'affiche pas les fichiers d'un dossier, si celui-ci n'est pas encore suivi.

Exercice

Ajouter des fichiers dans votre dossier et la commande git status : constater que le dossier apparaît (mais pas les fichiers)

Ignorer des fichiers

On peut vouloir ignorer des fichiers et des dossiers.

Exemple :

- fichiers de configs à la bdd
- dossiers d'uploads de fichiers
- dossiers de dépendance
- dossiers de build
- etc

Pour Ignorer des dossiers ou des fichiers, on crée un fichier **.gitignore** dans lequel on renseigne les chemins tous les dossiers / fichiers qui ne seront pas suivi.

Ex :

```
config.yml
*.log
logs/
/vendor/
```

Syntaxe : https://www.atlassian.com/fr/git/tutorials/saving-changes/gitignore

Exercice

Imaginons une application qui permettra d'uploader des photos. Cette application sera développée et testée sur un serveur local. Les images uploadées ne devront pas être commitées. Seuls les fichiers de code doivent être commités : html, css, js ...
Dans un nouveau répertoire, iInitialiser git, créer un fichier .gitignore et renseigner le.

Solution 1 :

Dans le dossier uploads, créer un fichier .gitkeep et ajouter ces lignes dans le fichier .gitignore :

```
/uploads
!.gitkeep
```

Solution 2 :

```
!/uploads
/uploads/*
```

Problèmes et solutions

- **« Je ne sais pas ce qu'il y a dans l'index ! »**

  Lister les fichiers indexés

  Cette commande permet de lister les fichiers placés dans l'index :

  ```
  git ls-files <fichier>
  ```

  https://git-scm.com/docs/git-ls-files

- « Je veux retirer des fichiers indexés par erreur! »

  Retirer des fichiers de l'index

  Cette commande permet de retirer des fichiers de l'index :

  ```
  git rm --cached index.php
  ```

  https://git-scm.com/docs/git-rm/fr

Astuces

Changer d’éditeur par défaut

Pour les fusions, l’éditeur par défaut est Vim. Pour utiliser Nano :

```
git config --global core.editor "nano"
```