GIT

[https://www.jedha.co/blog/git-et-github-definitions-differences-utilite#:~:text=D%C3%A9finition%20de%20Git&text=Ce%20type%20de%20logiciel%20est,besoin%2C%20%C3%A0%20une%20ancienne%20version.](https://www.jedha.co/blog/git-et-github-definitions-differences-utilite#:~:text=D%C3%A9finition%20de%20Git&text=Ce%20type%20de%20logiciel%20est,besoin%2C%20%C3%A0%20une%20ancienne%20version)

<https://git-scm.com/downloads>

<https://github.com/>

Bon lien : <https://git-scm.com/downloads/win>

Vidéos utiles : <https://www.youtube.com/watch?v=SWYqp7iY_Tc>

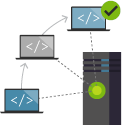
Vidéos utiles : <https://www.youtube.com/watch?v=8JJ101D3knE&t=1s>

**Pour commencer avec Git**

Nous pouvons imaginer que le design web est comme un bâtiment et Git est l'un des nombreux piliers importants sur lesquels le design web est construit. Nos principaux objectifs pour l'instant sont les suivants :

* Comprendre ce qu'est Git
* Apprendre à utiliser les commandes de base de Git
* Découvrir comment configurer Git
* Comprendre ce qu'est Github et comment travailler avec des dépôts distants

**Qu'est-ce qu'un VCS ?**

Un système de contrôle de version (VCS) est un outil qui permet aux équipes de développeurs de suivre et de gérer les modifications du code source. Il permet de prendre des instantanés des fichiers, sauvegardant ainsi les progrès de manière permanente. Sans VCS, les développeurs risquent de perdre des mois ou des années de travail en gardant plusieurs copies de code. Les VCS offrent un historique clair des modifications, permettant de revenir à des versions antérieures et d'explorer d'autres approches.  
  
  
  
  


**Qu'est-ce que Git ?**

Git est un VCS libre et gratuit conçu pour gérer tous les projets, des plus petits aux plus grands, avec rapidité et efficacité.

En clair, Git est un outil qui permet aux développeurs de suivre les versions de leur code au fil du temps. Pour ce faire, il crée des "instantanés" de l'état actuel de la base de code chaque fois que vous le lui demandez.

**Ces ressources peuvent vous aider**

**Les bases de Git**

[https://www.freecodecamp.org/news/learn-the-basics-of-git-in-under-10-minutes-da548267cc91/](https://www.freecodecamp.org/news/learn-the-basics-of-git-in-under-10-minutes-da548267cc91/" \t "_blank)

## Installation de Git

Git n’est généralement pas configuré par défaut sur votre ordinateur, vous devez donc l’installer et le configurer avant de pouvoir commencer à l’utiliser pour gérer votre code. Il est important de maintenir Git à jour, comme tous les autres logiciels de votre machine. Les mises à jour vous protègent contre les failles de sécurité, corrigent les bogues et vous donnent accès à de nouvelles fonctionnalités.

La méthode recommandée pour l’installation et la maintenance de Git est fournie ci-dessous pour les trois principales plateformes :

**Windows:** Téléchargez et installez Git pour [Windows](https://git-scm.com/download/win). Une fois installé, vous pourrez utiliser Git à partir de l’invite de commande ou de PowerShell. Nous vous recommandons de vous en tenir aux valeurs par défaut sélectionnées lors de l’installation, à moins que vous n’ayez une bonne raison de les modifier.  
  
  
**Linux :**Dans le terminal, il suffit de lancer sudo apt install git-all.  
  
  
  
**Mac :** La meilleure chose à faire est d’installer [Homebrew](https://brew.sh/) puis, à partir du terminal, de lancer la commande brew install git.

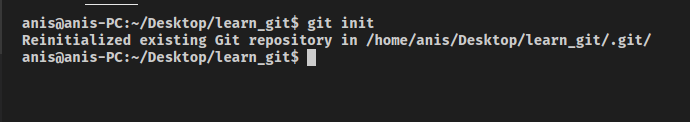
## Commencer à utiliser Git

L’utilisation de Git est très simple.

1. Accéder au dossier désiré en utilisant le terminal (prompt cmd).
2. Pour initialiser un dépôt, il suffit de lancer :

git init

Sortie



***P.S****: Cette commande vient de créer un dossier caché appelé .git, où toute la magie se produit.*

## Ajouter et valider des fichiers

Vous vous souvenez du dossier que nous avons créé ? Il est temps de l’utiliser. Nous allons créer un fichier et le placer dans ce dossier N’hésitez pas à ajouter ce que vous voulez !

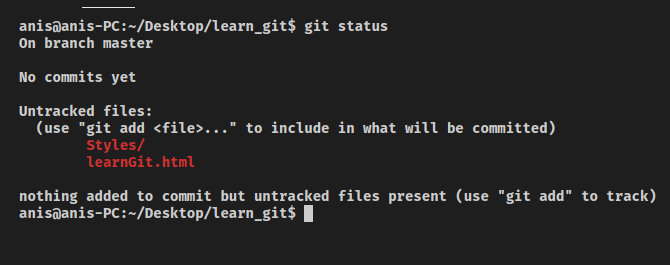
## Vérifier le statut de votre dépôt

Maintenant que nous avons quelques fichiers dans notre dépôt, voyons comment Git les traite. Pour vérifier l’état actuel de votre dépôt, nous utilisons la commande git status.

La commande git status permet de vérifier l’état actuel de votre dépôt.

git status

Sortie

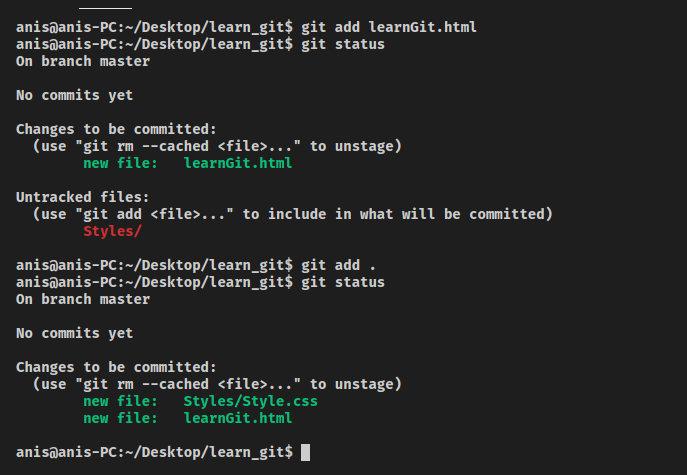


## Ajouter des fichiers à suivre pour Git

A ce stade, nous n’avons pas de fichiers à suivre pour Git. Nous devons ajouter des fichiers spécifiquement à Git afin de lui demander de les suivre. Nous ajoutons des fichiers à l’aide de la commande « add ». Après avoir exécuté git add . Git ajoutera tous les fichiers du dépôt dans une zone intermédiaire appelée **staging area**. Nous pouvons également ajouter ce que nous voulons simplement en exécutant git add monNomDeFichier.

git add .

Sortie



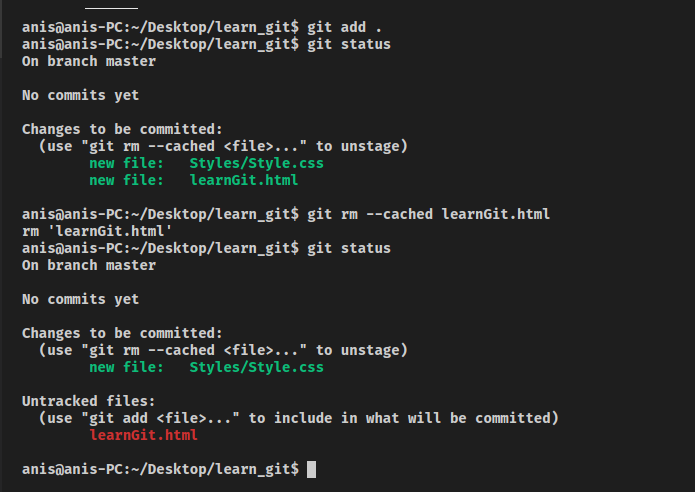
## Suppression de fichiers

Disons que vous avez ajouté des fichiers à Git et que vous ne voulez pas qu’il les suive. Dans une telle situation, vous devez demander à Git d’arrêter de les suivre. Cependant, l’exécution d’une simple commande git rm ne le supprimera pas seulement de Git, mais aussi de votre système de fichiers local ! Pour demander à Git d’arrêter de suivre un fichier tout en le conservant dans votre système local, exécutez la commande suivante :

La commande Git rm permet de supprimer un fichier de votre système local.

git rm --cached [nom\_du\_fichier]

Sortie



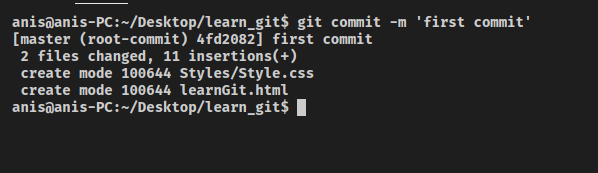
## Commissionner les modifications

Une fois que vous avez mis en scène vos fichiers, vous pouvez les valider dans Git. Imaginez une commande de validation comme un instantané à un moment donné, auquel vous pouvez revenir pour accéder à votre référentiel à ce stade. Vous attribuez un message de validation à chaque validation, que vous pouvez transmettre avec le préfixe -m.

Les messages de validation sont des messages de validation.

git commit -m 'premier commit'

Sortie



**Ces ressources peuvent vous aider**

**Les commandes basiques git**

[https://dzone.com/articles/top-20-git-commands-with-examples](https://dzone.com/articles/top-20-git-commands-with-examples" \t "_blank)