



insy2s



git

Qu'est-ce que Git ?

Git est un logiciel de versioning créé en 2005 par Linus Torvalds, le créateur de Linux.

Un logiciel de versioning, ou logiciel de gestion de versions est un logiciel qui permet de conserver un historique des modifications.

Il permet de rapidement identifier les changements effectués et de revenir à une ancienne version en cas de problème.

Les logiciels de gestion de versions sont quasiment incontournables aujourd'hui car ils facilitent grandement la gestion de projets et ils permettent de travailler en équipe de manière beaucoup plus efficace.

Parmi les logiciels de gestion de versions, Git est le leader incontesté et il est donc indispensable pour tout développeur de savoir utiliser Git.

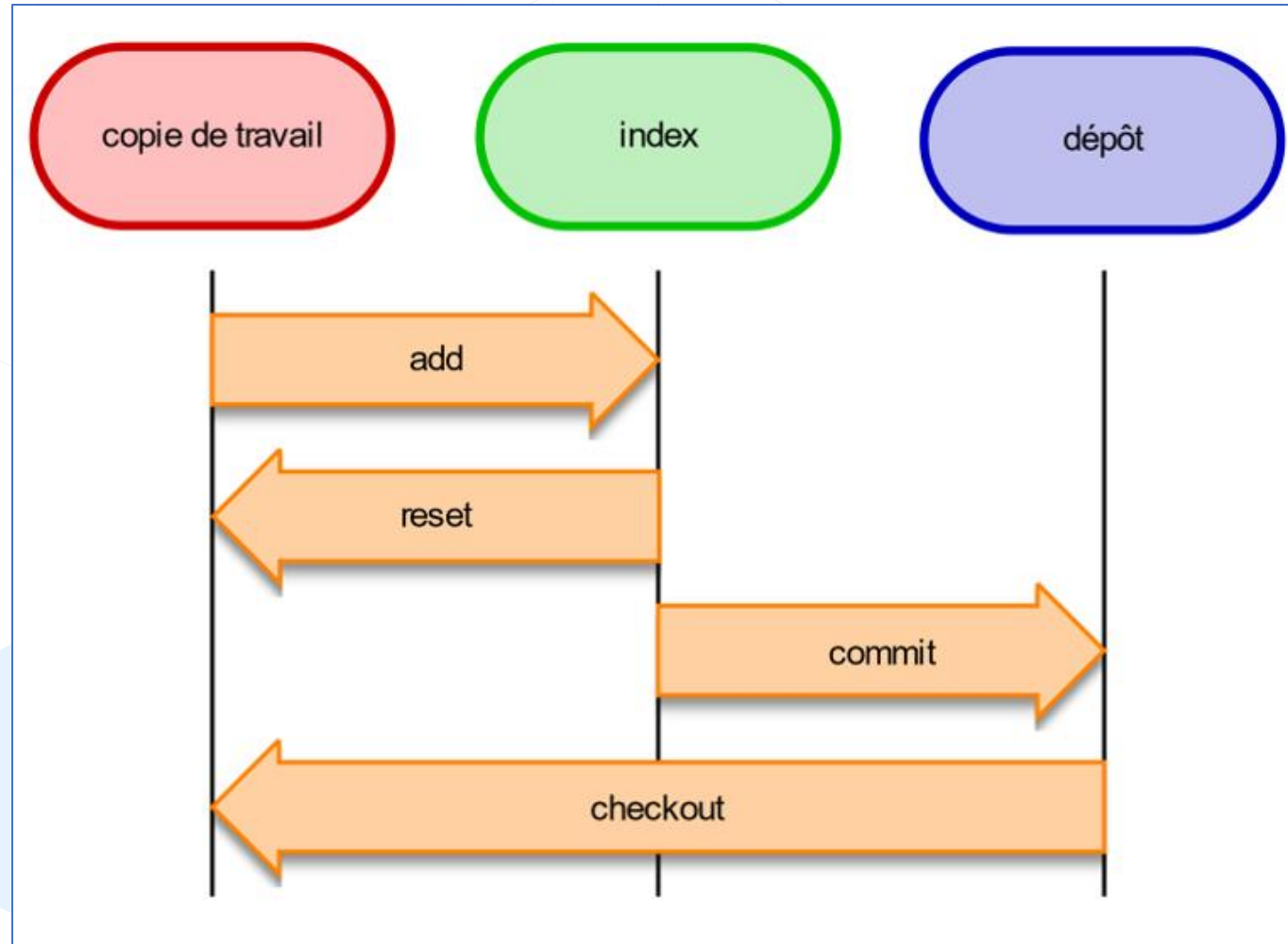
Qu'est-ce que Git ?

Git est un logiciel de gestion de versions.

Git va nous permettre d'enregistrer les différentes modifications effectuées sur un projet et de pouvoir retourner à une version précédente du projet.

La copie de l'intégralité des fichiers d'un projet et de leur version située sur le serveur central est appelé un dépôt. Git appelle également cela "repository" ou "repo" en abrégé.

Qu'est-ce que Git ?

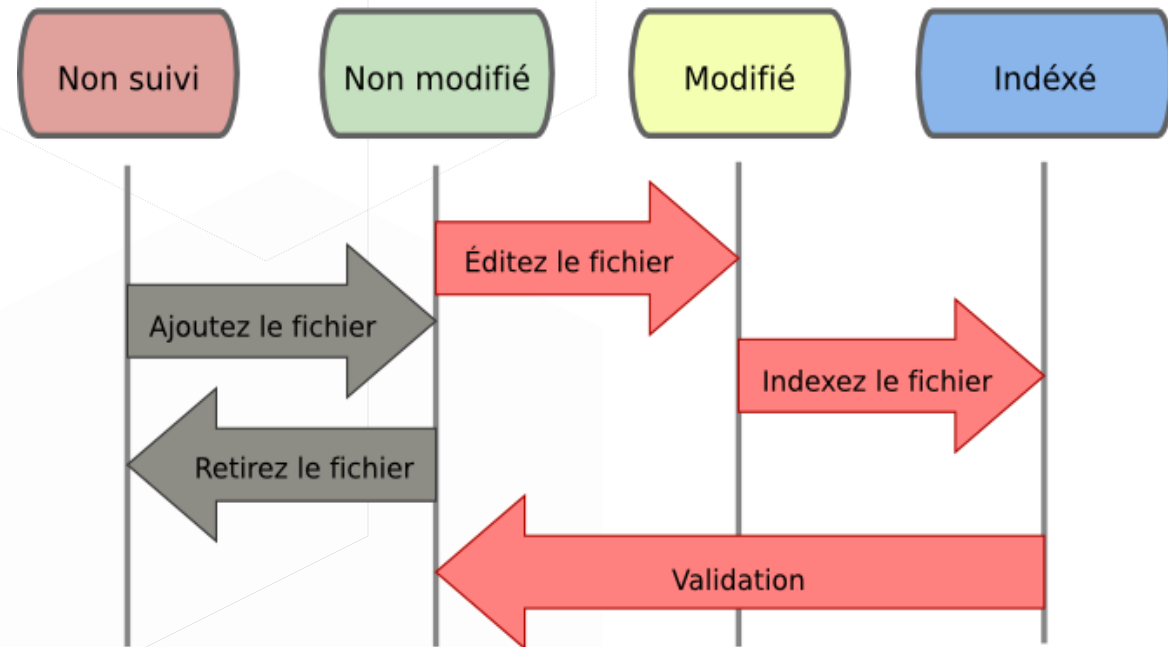


Enregistrer des modifications dans un dépôt

Quatre états d'un projet Git:

- **Non suivi**: fichier n'étant (n'appartenant) pas ou plus géré par Git;
- **Non modifié**: fichier sauvegardé de manière sûre dans sa version courante dans la base de données du dépôt;
- **Modifié**: fichier ayant subi des modifications depuis la dernière fois qu'il a été soumis;
- **Indexé**: idem pour modifié, sauf qu'il sera pris instantané dans sa version courante de la prochaine soumission (commit).

Cycle d'un fichier



Modèle centralisé vs Modèle décentralisé :

Les logiciels de gestion de version sont construits sur deux modèles :

- le modèle centralisé
- le modèle décentralisé encore appelé modèle distribué.

Le modèle centralisé :

la source du code du projet est hébergé sur un serveur distant central et les différents utilisateurs doivent se connecter à ce serveur pour travailler sur ce code.

Le modèle distribué :

le code source du projet est toujours hébergé sur un serveur distant mais chaque utilisateur est invité à télécharger et à héberger l'intégralité du code source du projet sur sa propre machine.

Qu'est ce qu'une Branche d'un dépôt ?

Créer une branche, c'est créer une "copie" de votre projet à partir d'un point donné pour développer et tester de nouvelles fonctionnalités sans impacter le projet de base.

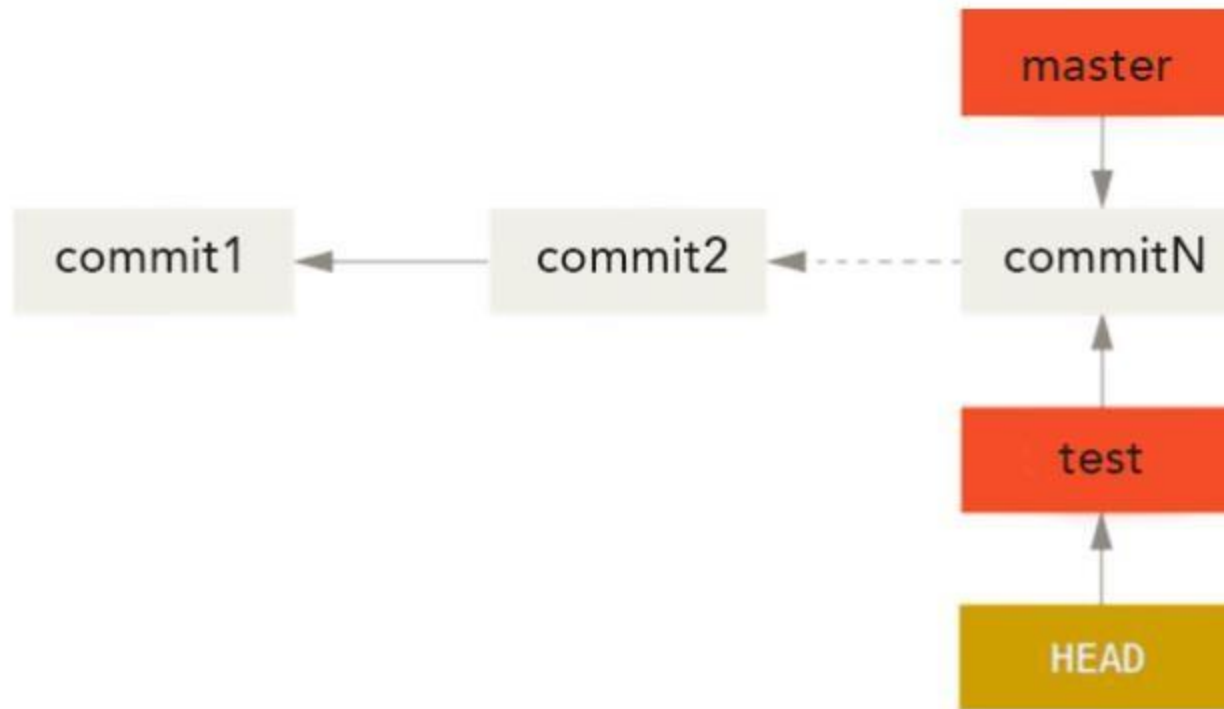
La branche par défaut dans Git quand vous créez un dépôt s'appelle master et elle pointe vers le dernier des commits réalisés.



Qu'est ce qu'une Branche d'un dépôt ?

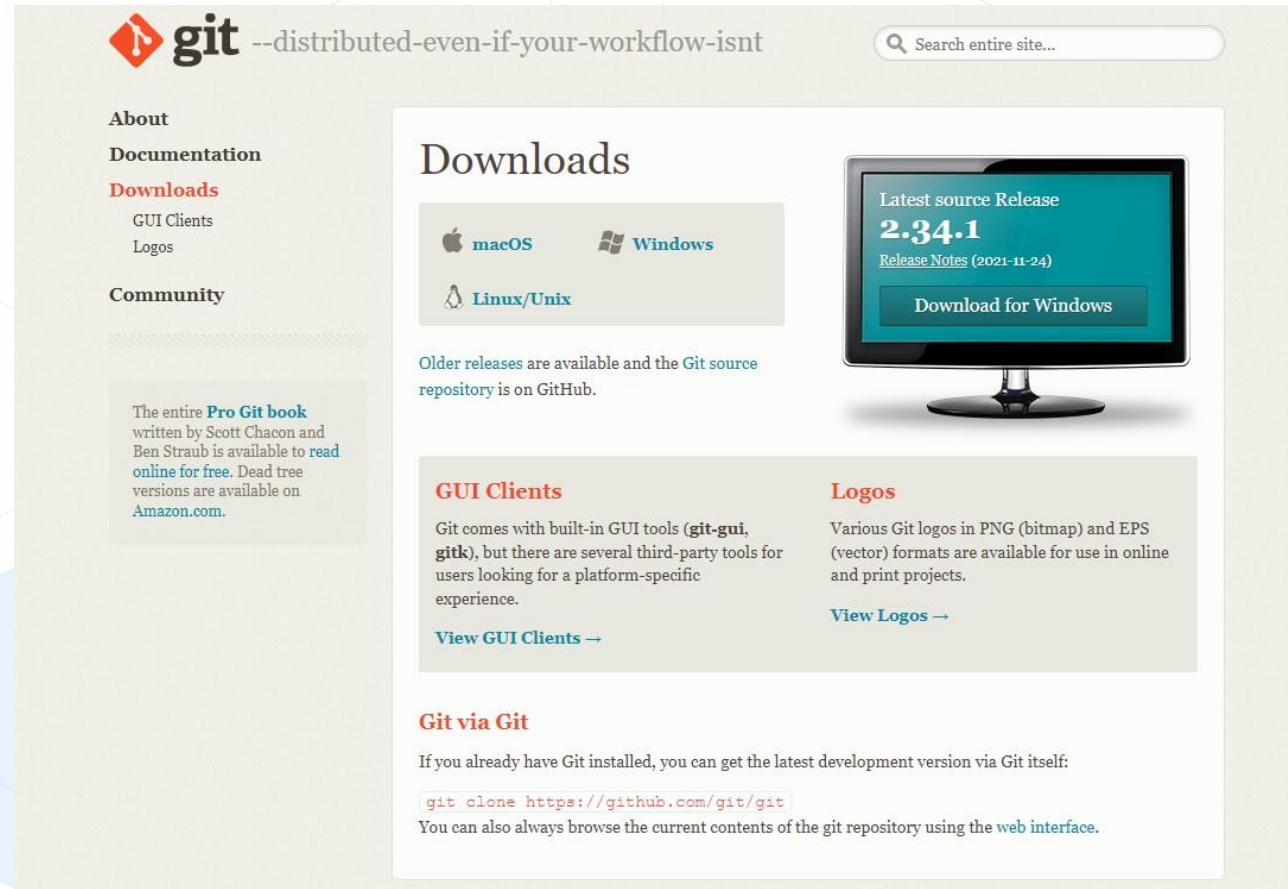
Pour déterminer quel pointeur vous utilisez, c'est-à-dire sur quelle branche vous vous trouvez, Git utilise un autre pointeur spécial appelé HEAD.

HEAD pointe sur la branche Master par défaut. La commande git branch permet de créer une nouvelle branche mais ne déplace pas le pointeur HEAD



Comment installer Git ?

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



The screenshot shows the Git website's Downloads page. The header features the Git logo and the tagline "--distributed-even-if-your-workflow-isnt", along with a search bar. The left sidebar contains links for "About", "Documentation", "Downloads" (highlighted), "GUI Clients", "Logos", and "Community". The main content area is titled "Downloads" and includes a section for the "Latest source Release" (2.34.1) with a "Download for Windows" button. Below this, there are links for "Older releases" and the "Git source repository". The page also features sections for "GUI Clients" and "Logos".

git --distributed-even-if-your-workflow-isnt

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloads

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.34.1
Release Notes (2021-11-24)
[Download for Windows](#)

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).

Qu'est-ce que GitHub ?

GitHub est un service en ligne qui permet d'héberger des dépôts ou repo Git. C'est le plus grand hébergeur de dépôts Git du monde.

Une grande partie des dépôts hébergés sont publics, ce qui signifie que n'importe qui peut télécharger le code de ces dépôts et contribuer à leur développement en proposant de nouvelles fonctionnalités.

Pour récapituler :

- Git est un logiciel de gestion de version
- GitHub est un service en ligne d'hébergement de dépôts Git qui fait office de serveur central pour ces dépôts.

<https://github.com/>

Commande GIT de base

Git config :

L'une des commandes git les plus utilisées est **git config**. On l'utilise pour configurer les préférences de l'utilisateur : son mail le nom d'utilisateur et le format de fichier etc. Par exemple, la commande suivante peut être utilisée pour définir le mail d'un utilisateur:

```
$ git config --global user.email sam@google.com
```

Git Clone :

La **commande git clone** est utilisée pour la vérification des dépôts. Si le dépôt se trouve sur un serveur distant :

```
$ git clone alex@93.188.160.58:/chemin/vers/dépôt
```

Commande GIT de base

Git add

La **commande git add** peut être utilisée pour ajouter des fichiers à l'index. Par exemple, la commande suivante ajoutera un fichier nommé temp.txt dans le répertoire local de l'index:

```
$ git add temp.txt
```

Pour ajouter tout les fichiers à l'index nous pouvons utiliser l'option **-A** :

```
$ git add -A
```

Ou le point

```
$ git add . (at the root of your project folder)
```

Commande GIT de base

Git commit :

La **commande git commit** permet de **valider les modifications apportées** au HEAD. Notez que tout commit ne se fera pas dans le dépôt distant.

```
$ git commit -m "Description du commit"
```

Git status :

La **commande git status** affiche la liste des fichiers modifiés ainsi que les fichiers qui doivent encore être ajoutés ou validés.

```
$ git status
```

Commande GIT de base

Git Branche :

La **commande git branch** peut être utilisée pour répertorier, créer ou supprimer des branches.

Pour répertorier toutes les branches présentes dans le dépôt :

```
$ git branch
```

Pour supprimer une branche :

```
$ git branch -d <nom-branche>
```

Commande GIT de base

Git checkout :

La **commande git checkout** peut être utilisée pour créer des branches ou pour basculer entre elles.

Par exemple nous allons créer une branche et basculer sur cette branche :

```
$ git checkout -b <nom-branche>
```

Pour passer simplement d'une branche à une autre, utilisez :

```
$ git checkout <nom-branche>
```

Commande GIT de base

Git Pull :

Pour fusionner toutes les modifications présentes sur le dépôt distant dans le répertoire de travail local, la commande pull est utilisée.

```
$ git pull
```

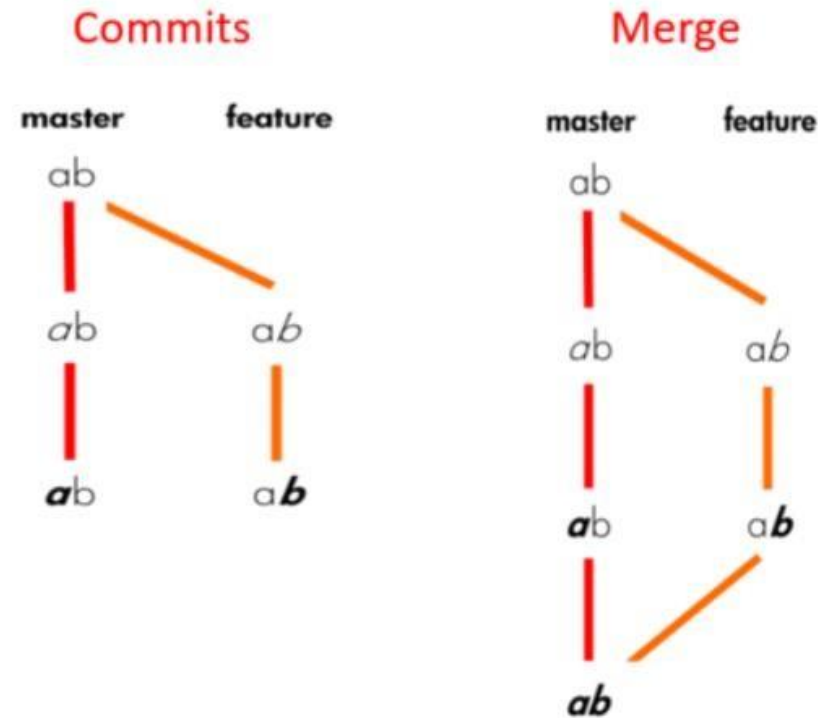
Git merge :

La commande `git merge` est utilisée pour fusionner une branche dans la branche active.

```
$ git merge <nom-branche>
```


Commande GIT de base

Git merge :



Commande GIT de base

Git Diff :

La commande git diff permet de lister les conflits. Pour visualiser les conflits d'un fichier, utilisez :

```
$ git diff --base <nom-fichier>
```

La commande suivante est utilisée pour afficher les conflits entre les branches à fusionner avant de les fusionner :

```
$ git diff <branche-source> <branche-cible>
```

Pour simplement énumérer tous les conflits actuels, utilisez :

```
$ git diff
```

Git rm :

Git rm peut être utilisé pour supprimer des fichiers de l'index et du répertoire de travail :

```
$ git rm <nom-fichier>
```

<https://www.julienkrier.fr/articles/git-cheat-sheet>