# Méthode et Outil de versionning : Git

Histoire de Git :

Créé en 2005 par Linys Torvalds lors du développement du Linux Kernel, Git est un puissant **outil de versionning**. Le rôle principal de Git est de permettre plusieurs collaborations sur le même projet à travers plusieurs plateformes comme GitHub, GitLab, BitBuckett, etc.

Git propose un certain nombre de commandes s’exécutant côté client et aussi côté serveur, il est maintenant intégré dans de nombreux IDE (JetBrains, Visual Studio Code, etc.).

Installation de l’outil Git :

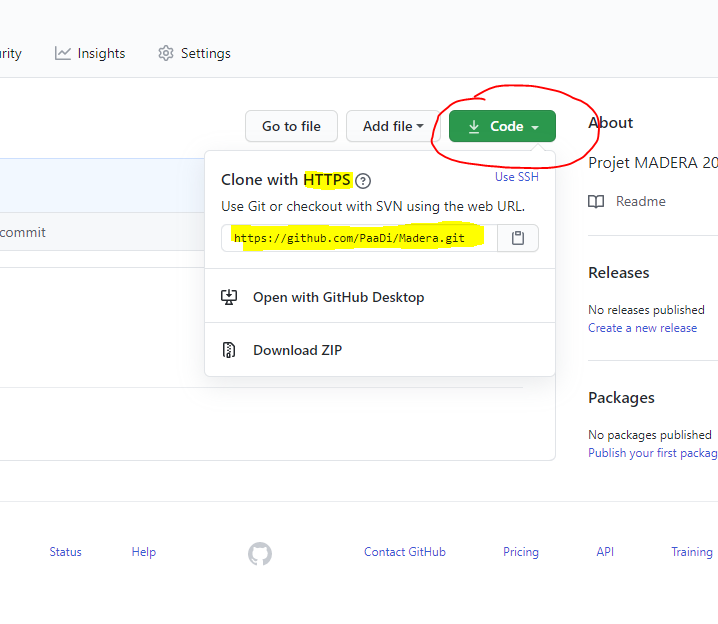
Version Windows : <https://git-for-windows.github.io/>

Version Linux : <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

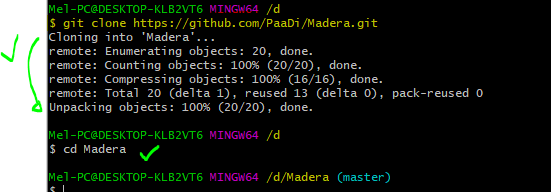
Version Mac : <https://git-scm.com/download/mac>

Initialisation du projet :

1. Ouvrir l’**outil Git**.
2. Via de basique commandes CMD (cd, mkdir), se placer dans le répertoire où nous souhaitons cloner le projet :
3. Dans GitHub, **Cliquer sur Code** et **copier l’URL du projet** :



1. Saisir la commande **git clone** [**https://github.com/PaaDi/Madera.git**](https://github.com/PaaDi/Madera.git) pour cloner le projet :



Ci-dessus, nous voyons que le projet a correctememnt était cloné, avec la commande **cd Madera**, nous pouvons maintenant accéder au repository. Nous serons automatiquement sur la branche master dès lors du premier clonage.

## Organisation des branches

Durant le développement, il sera important de pousser les premières modifications sur des **branches tierces** à la branche master.

La branche **master** nous servira de version officielle. Les branches nous permettent de diverger de cette branche master et continuer à travailler sans impacter cette ligne.



La manière dont Git gère les branches est incroyablement légère et permet de réaliser des opérations sur les branches quasi instantanée, et de basculer sur une autre branche très facilement.

Il est important d’utiliser cette fonctionnalitée Git qui est une des fonctionnalitées les plus intéréssantes pour le travail en **collaboration**.

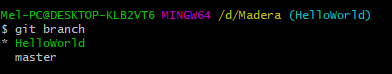
Règle fondamentale : Ne jamais effectuer de modification directememnt sur la branche master.

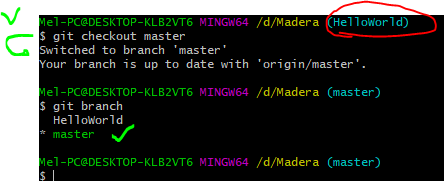
# Processus d’une routine Git

1. Créer une nouvelle branche avec la commande **git checkout -b [NomBranche]**:

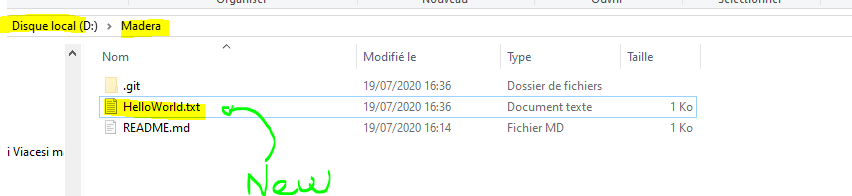


*Pour consulter les branches actuellement disponible, il suffit d’utiliser la commande git branch qui affiche en vert la branche sur laquelle nous sommes* :

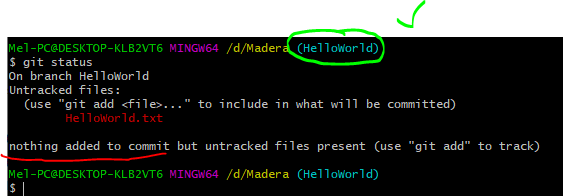


1. La commande **git checkout [NomBranche]**, nous permettra de changer de branche :
2. Une fois placé sur la bonne branche, nous pourrons développer tranquillement et faire des tests sans impacter les autres, git permet de gérer les conflits.

*Pour exemple, j’ai ajouté au projet un nouveau fichier « HelloWorld.txt » que l’on va pousser jusqu’à la branche master.*

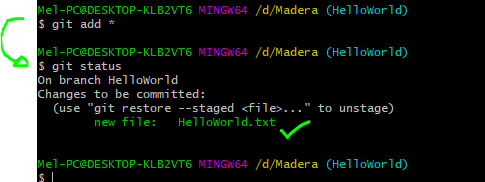


1. La commande git status nous permettra de connaître les différentes modifications apportées au repository :



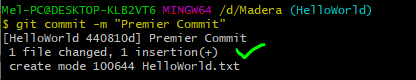
*Ci-dessus, nous voyons bien que nous sommes sur la branche HelloWorld, et qu’un fichier HelloWorld.txt a été ajouté. M malgès l’ajout, le fichier ne sera pas automatiquement dans le commit, il faudra ajouter au « panier » la modification pour éffectuer en suivant le commit.*

1. Pour ajouter la modification à notre commit, il suffit de taper la commande **git add \*** ou **git add [NomFichier]** :

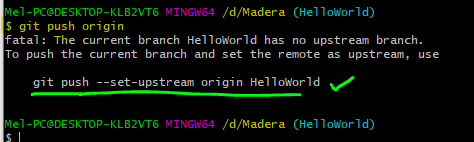


*En tapant la commande git status après avoir ajouté la modification au panier, nous voyons que celui-ci passe au vert. Nous sommes donc prêt à envoyer le commit.*

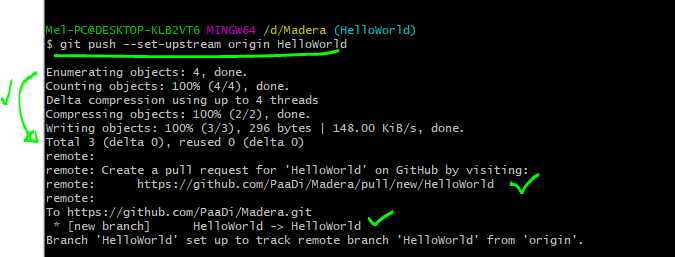
1. La commande **git commit -m "brève description du commit"** nous permettra d’envoyer le commit sur la branche.



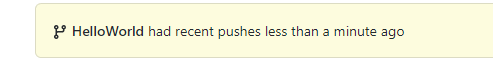
1. Pour envoyer la modification à l’origine (Disponible à tous sur GitHub), il faudra saisir la commande **git push origin**:

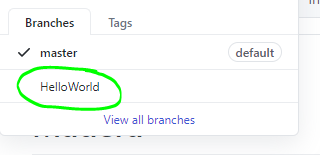


1. Git nous propose une commande à saisir pour pousser la modification, copier puis saisissez la :

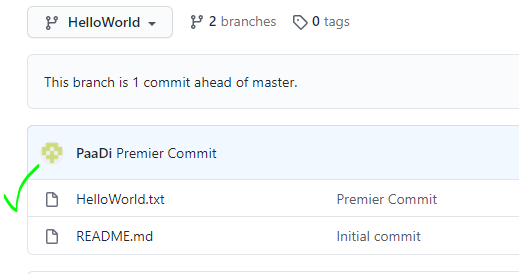


*Ci-dessus, nous voyons qu’une nouvelle branche HelloWorld a été crée sur l’origine, elle sera maintenant accessible depuis le GitHub dans l’onglet branche :*

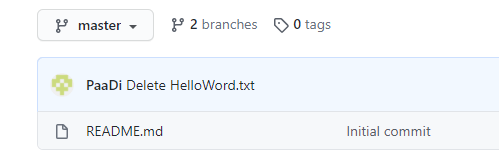




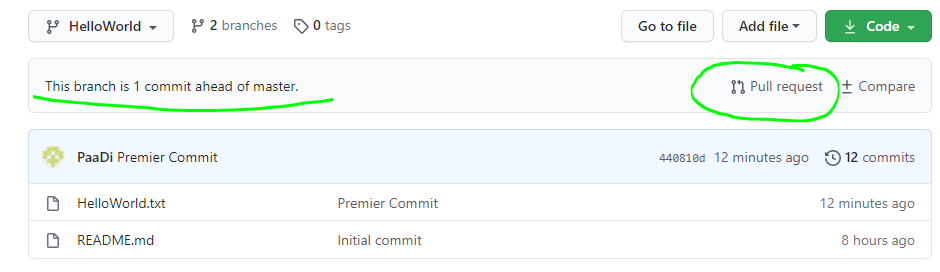
Nous voyon bien que **le nouveau fichier est disponible sur la branche HelloWorld** :



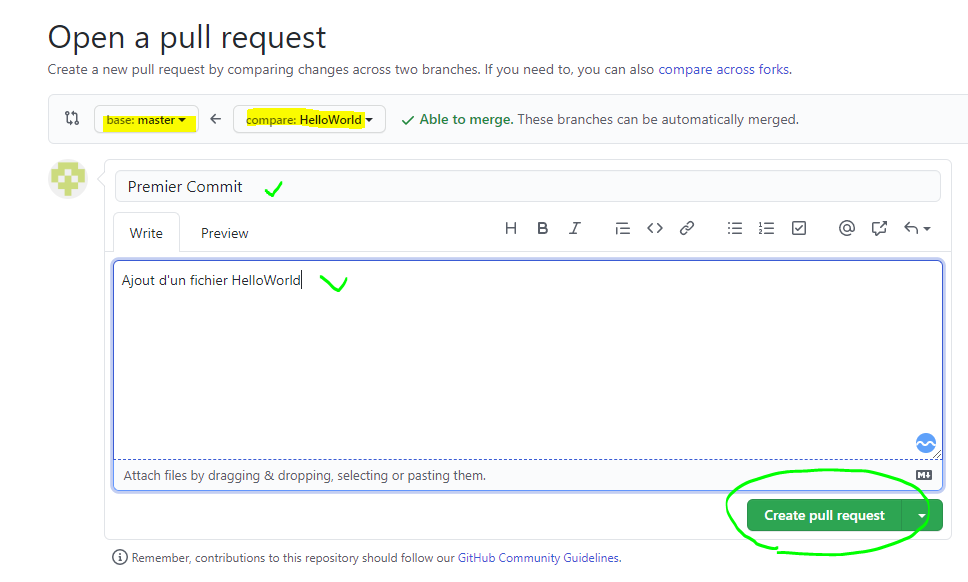
Mais **pas encore sur la branche Master** :



1. En allant sur la branche HelloWorld, nous voyons qu’**un commit est disponible**. Pour pouvoir pousser la modification sur la branche master, nous devrons passer par un **Pull Request** qui permettra de valider la modification avant le push (étape importante) :

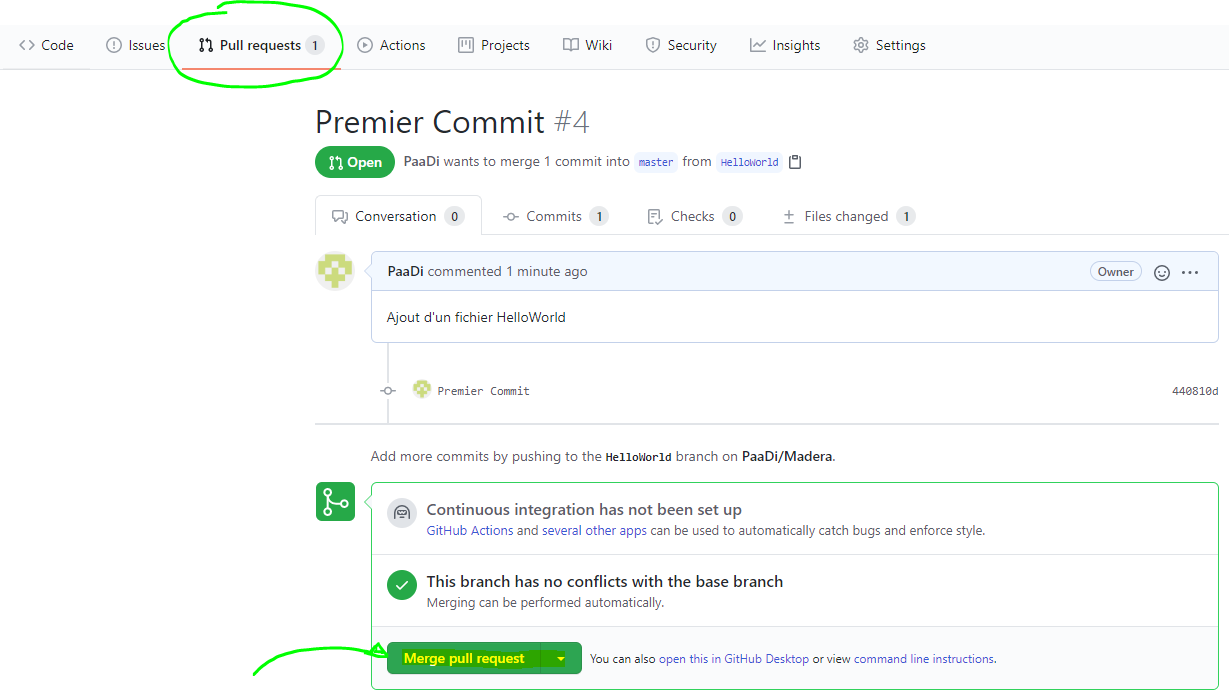


1. En cliquant sur **Pull Request**, il faudra **saisir un titre** à notre commit ainsi qu’**une description** qui nous permettrons de tracer les différentes modifications apportées par les membres du groupe projet. Cliquer sur **Create Pull Request** une fois les champs remplis :

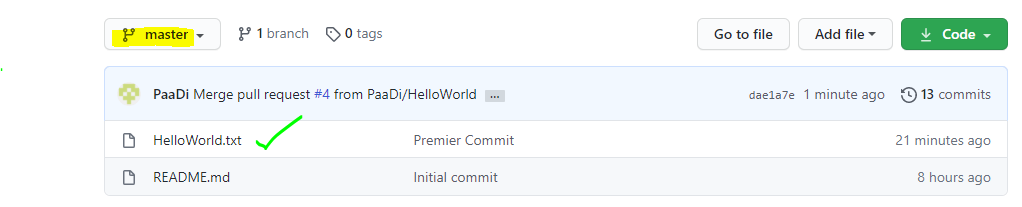


*A noter : Il est aussi possible de changer la branche sur laquelle nous voudrons merger le commit.*

1. Pour merger la requête vers la brnahce master, il faudra valider la Pull Request en allant dans l’onglet Pull Requests, et cliquer sur Merge Pull Request pour valider et pousser la modification :



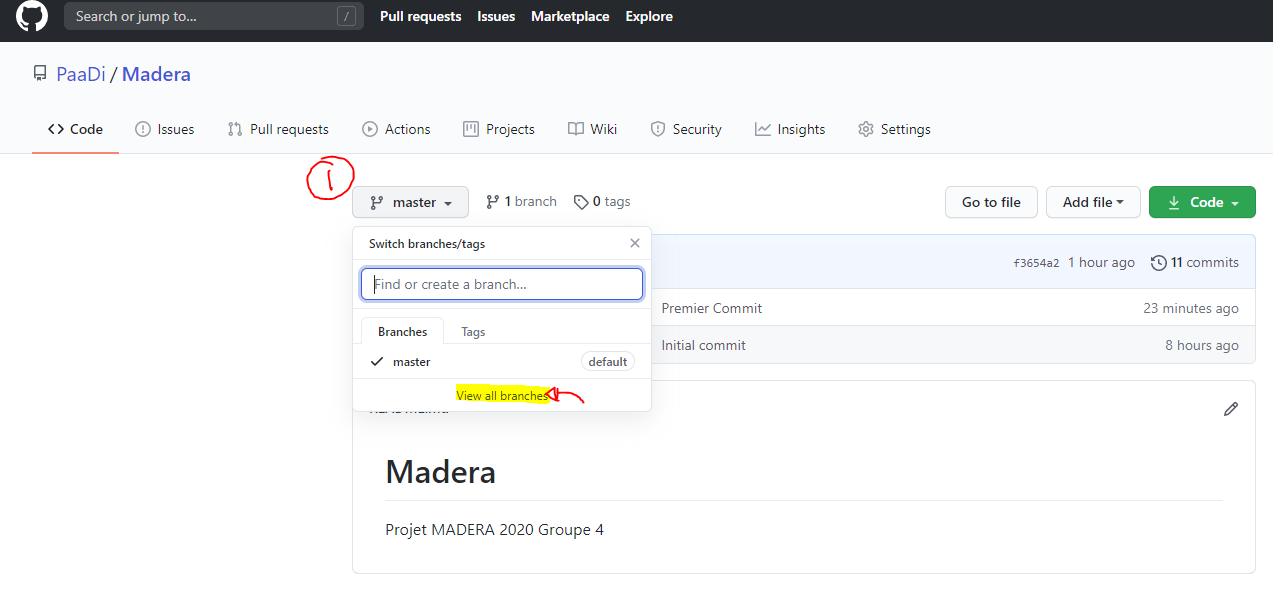
1. Le fichier est maintenant **disponible** sur la branche master :



## Bonnes Pratiques :

1. Supprimer une branche :

*Cliquer sur la poubelle dans GitHub*



# Si la branch est local et n'est pas créée sur le repo distant

git branch -d nom\_de\_ma\_branch\_local

# Si la branch est présente sur le repo distant

git push origin --delete nom\_de\_ma\_branch\_distante

1. Mettre à jour le dépôt local (Projet MADERA) :

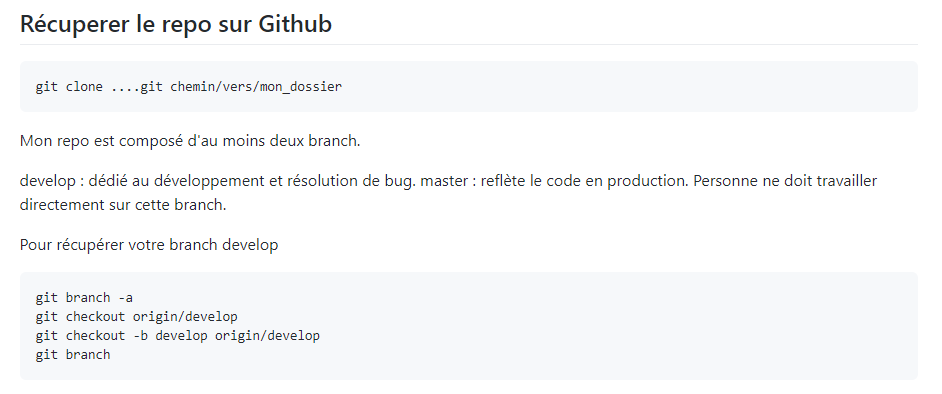
git pull

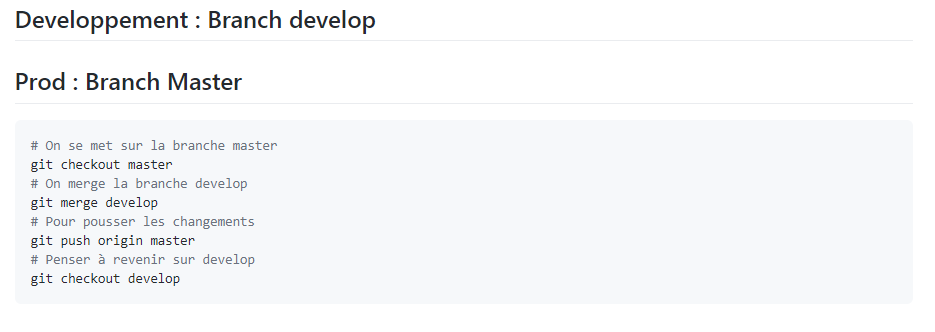
1. Annuler le dernier commit :

git reset HEAD^

Pour indiquer à quel commit on souhaite revenir, il existe plusieurs notations :

1. HEAD : dernier commit ;
2. HEAD^ : avant-dernier commit ;
3. HEAD^^ : avant-avant-dernier commit ;





## Liens utiles :

Liste commandes basiques Git : <https://gist.github.com/aquelito/8596717>

Formation Git Grafikart (GRATUIT), 18 vidéos : <https://www.youtube.com/watch?v=rP3T0Ee6pLU&list=PLjwdMgw5TTLXuY5i7RW0QqGdW0NZntqiP>

Documentation Officielle Git : <https://git-scm.com/doc>

OpenClassroom Git : <https://openclassrooms.com/fr/courses/1233741-gerez-vos-codes-source-avec-git>