Comment utiliser Git pour la première fois

.

GIT et GitHub

Marc Maffettone 2017

Sommaire

Table des matières

Les commandes GIT

git init : Initialise un nouveau dépôt Git. Jusqu'à ce que vous exécutiez cette commande dans un dépôt ou répertoire, c'est juste un dossier ordinaire. Seulement après avoir entré cette commande, il accepte les commandes Git qui suivent.

git config : raccourci de "configurer," ceci est tout particulièrement utile quand vous paramétrez Git pour la première fois.

git help : Oublié une commande ? Tapez-ça dans la ligne de commande pour afficher les 21 commandes les plus courantes de Git. Vous pouvez aussi être plus spécifique et saisir "git help init" ou tout autre terme pour voir comment utiliser et configurer une commande spécifique git.

git status : Vérifie le statut de votre repository. Voir quels fichiers sont à l'intérieur, quelles sont les modifications à *commiter*, et sur quelle branche du repository vous êtes en train de travailler.

git add : Ceci n'ajoute *pas* de nouveaux fichiers dans votre repository. Au lieu de cela, cela porte de nouveaux fichiers à l'attention de Git. Après avoir ajouté des fichiers, ils sont inclus dans les "instantanés" du dépôt Git.

git commit : la commande la plus importante de Git. Après avoir effectué toute sorte de modification, vous entrez ça afin de prendre un “instantané” du dépôt. Généralement cela s’écrit sous la forme git commit -m “Message ici“. Le -m indique que la section suivante de la commande devrait être lue comme un message.

git branch : Vous travaillez avec plusieurs collaborateurs et vous voulez produire des modifications de votre côté ? Cette commande vous permet de construire une nouvelle branche, ou une chronologie des commits, des modifications et des ajouts de fichiers qui sont complètement les vôtres. Votre titre va après la commande. Si vous vouliez créer une nouvelle branche appelée “chats”, vous saisiriez git branch chats.

git checkout : Permet littéralement de vérifier un dépôt dans lequel vous n’êtes pas. C’est une commande de navigation qui vous permet de vous déplacer vers le répertoire que vous voulez vérifier. Vous pouvez utiliser cette commande sous la forme git checkout master pour regarder la branche master, ou git checkout chats pour regarder une autre branche.

git merge : Lorsque vous avez fini de travailler sur une branche, vous pouvez fusionner vos modifications vers la branche master, qui est visible pour tous les collaborateurs. git merge chats prendrait toutes les modifications que vous avez apportées à la branche “cats” et les ajoutera à la la branche master.

git push : Si vous travaillez sur votre ordinateur local, et voulez que vos commits soient visibles aussi en ligne sur Github, vous “push”ez les modifications vers Github avec cette commande.

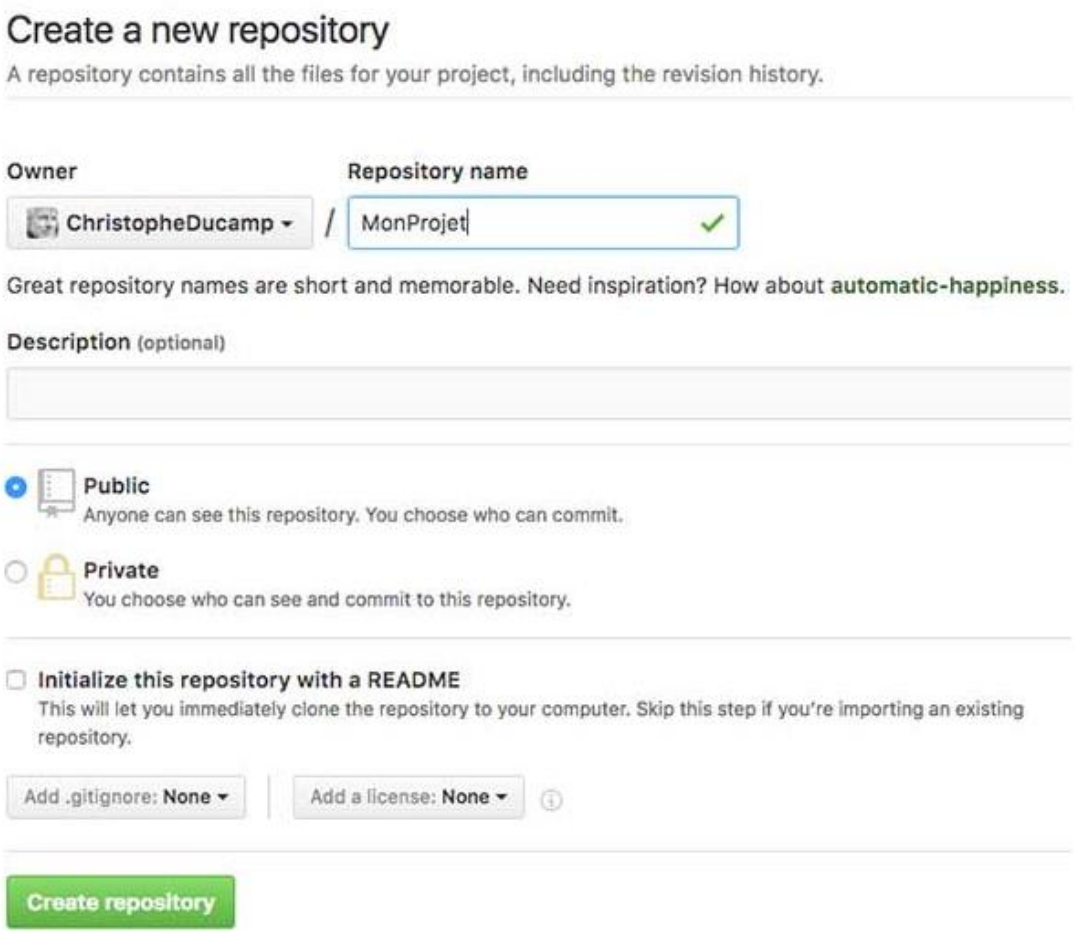
git pull : Si vous travaillez sur votre ordinateur local, et que vous voulez la version la plus à jour de votre repository pour travailler dessus, vous “pull”ez (tirez) les modifications provenant de Github avec cette commande

Initier GIT et communiquer avec GitHub

1.Se définir auprès de GIT

```
git config --global user.name "Votre Nom Ici"
git config --global user.email "votre_email@votre_email.com"
```

2.Créer un repository sur GitHub



3.Créer un repository local

```
mkdir MonProjet
\\ Créera un fichier à l'emplacement voulu //
```

4. Initier Git

```
git init
\\ Va initier un fichier .git à la racine du dossier //
```

5. Mettre ses fichiers ou les crées

```
$ touch Readme.txt ( Exemple )
\\ Créer un fichier .txt //
```

6. Signaler à GIT qu'un ou des fichiers sont présents

```
$ git add Readme.txt
\\ Signal à GIT que le fichier est là //
```

7. Consigner son fichier

```
$ git commit -m "Ajout Lisez-moi.txt"
\\ Consigne le fichier. Toujours parler au présent //
```

8. Maintenant que nous avons nos fichiers ajoutés et « commités », nous devons definir notre route de communication avec GitHub.

```
$ git remote add origin https://github.com/Marc01750/TestGit.git

Ou :
Marc01750 = Le nom du compte GitHub
TestGit = Le nom du repository

\\ Ajoute la « route » dans la configuration git //
```

9. Pour confirmer que notre route est définie il suffit de taper dans la console

```
$ git remote –v


Si la route ci-dessus apparaît c’est que tout vas bien.


Ps supprimer une route : git remot rm NomDeLaRoute


\\ Pour afficher toutes les « routes » actuelles //
```

10. Envoyer les fichiers sur notre repository distant

```
$ git pull origin master

$ git fetch

$ git push origin master



\\ Avant de faire le premier push il faut impérativement effectuer un pull suivi
d’un fetch. Enfin finir par un push, qui envois vos fichiers sur GitHub. //
```

Sources

https://fr.wikibooks.org/wiki/Git/Synchroniser_le_d%C3%A9p%C3%B4t_local_avec_le_d%C3%A9p%C3%B4t_distant

https://www.christopheducamp.com/2013/12/16/github-pour-nuls-partie-2/