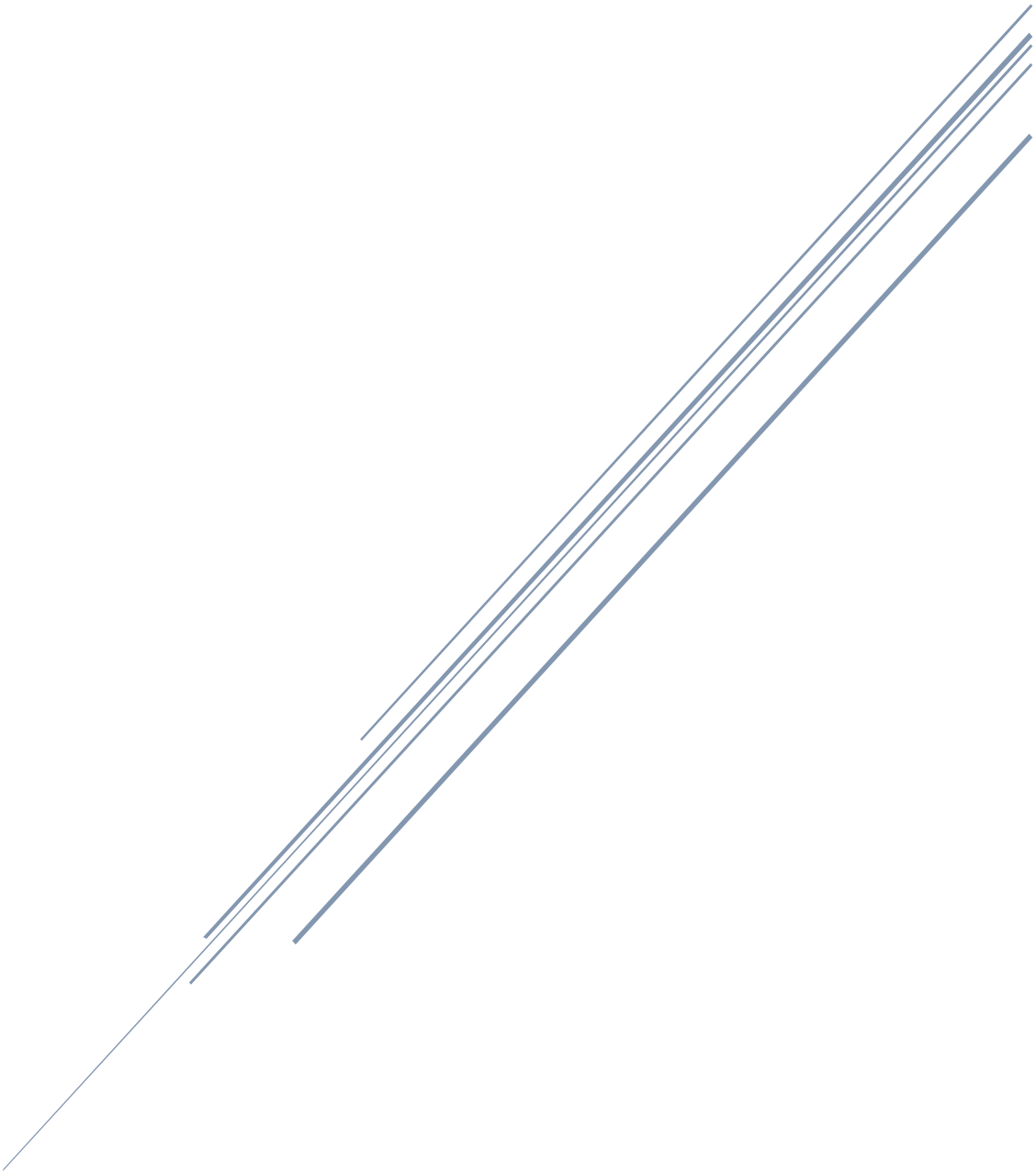
DOCUMENTATION GIT

Utilisation de l'outil de versionning GIT



Cédrik DODDE

A. Qu'est-ce que GitHub ?

GitHub est l'une des plus grandes communautés de développeurs au monde. C'est une plate-forme complexe qui favorise la collaboration et la communication entre les développeurs. GitHub possède certain nombre de fonctionnalités utiles qui permettent aux équipes de développement de travailler ensemble sur le même projet et de créer facilement de nouvelles versions de logiciels sans perturber les versions actuelles, mais cela ne s'arrête pas là.
Une fois que de nouveaux ajouts à un programme sont terminés, par exemple, ils peuvent facilement être intégrés à des programmes existants. GitHub rend également extrêmement simple de travailler ensemble sur des chaînes de code pour vraiment composer et perfectionner même les plus petites parties d'un programme. Avec GitHub, vous pouvez collaborer et travailler sur des projets avec d'autres partout dans le monde.

B. Quels sont les avantages de GitHub ?

La première est qu'il permet une collaboration et un contrôle de version simples et fluides. Cela vous permet de travailler sur du code avec n'importe qui de n'importe où. De plus, de nombreux employeurs utilisent GitHub. Donc, si vous envisagez de trouver un emploi, vous aurez l'air vraiment bien si vous connaissez déjà GitHub. GitHub est une plateforme d'apprentissage et de collaboration robuste.

C. Comment utiliser GitHub ?

GitHub est complexe, mais comprendre quelques notions de base vous aidera à démarrer. Pour utiliser GitHub, vous devez d'abord être en mesure de suivre ces quelques étapes.

- ✓ **Inscrivez-vous à GitHub**
  Pour utiliser GitHub, vous aurez besoin d'un compte GitHub. Vous pouvez créer un compte GitHub gratuit sur https://github.com/ et commencer à utiliser GitHub immédiatement. Avec un compte gratuit, vous aurez accès à un nombre illimité de dépôts publics et privés. Vous bénéficierez également de fonctionnalités de suivi des bogues et de gestion de projet. Le seul inconvénient est que vous n'aurez droit qu'à trois collaborateurs pour les dépôts privés.

- ✓ **Installez Git**
  GitHub fonctionne sur Git. Qu'est-ce que Git ?
  Git est un système de contrôle de version créé par l'icône de la programmation, Linus Torvald. Il a initialement créé Git pour suivre les modifications apportées au code source alors qu'il développait le système d'exploitation Linux. Git aide les programmeurs à collaborer, à coordonner le travail et à travailler ensemble sur des projets de code et de développement complexes. Git suit les changements et aide les équipes à travailler à distance sur des programmes complexes.
  En y regardant de plus près, Git s'avère être une collection d'exécutables en ligne de commande, conçus pour être exécutés dans un environnement de style Unix. C'est pourquoi il se sent chez lui, dès la sortie de la boîte, sur les systèmes d'exploitation Linux et macOS, car ils fournissent tous deux un terminal de ligne de commande intégré de style Unix.
  Cependant, Microsoft Windows n'inclut pas une telle ligne de commande de style Unix et utilise à la place l'invite de commande Windows. Mais Git, seul, ne peut pas fonctionner dans

cet environnement. C'est là qu'intervient Git Bash : c'est un package facile à installer qui apporte Git à Windows.
Le nom indique que "Git Bash" fournira à un utilisateur deux composants principaux :
(1) Git - La collection de programmes en ligne de commande qui constitue le système de contrôle de version Git.
(2) Bash - Le nom d'un shell par défaut populaire sur macOS et Linux.
Cela signifie que le package Git Bash installe non seulement Git, mais également le shell Bash et certains utilitaires importants pour Bash.

Installation sur Linux

Si vous utilisez Fedora (ou toute distribution basée sur RPM étroitement liée, telle que RHEL ou CentOS), vous pouvez utiliser dnf:
$ sudo dnf install git-all

Si vous êtes sur une distribution basée sur Debian, comme Ubuntu, essayez apt:

$ sudo apt install git-all

Pour plus d'options, il existe des instructions d'installation sur plusieurs distributions Unix différentes sur le site Web de Git, à l' adresse https://git-scm.com/download/linux .

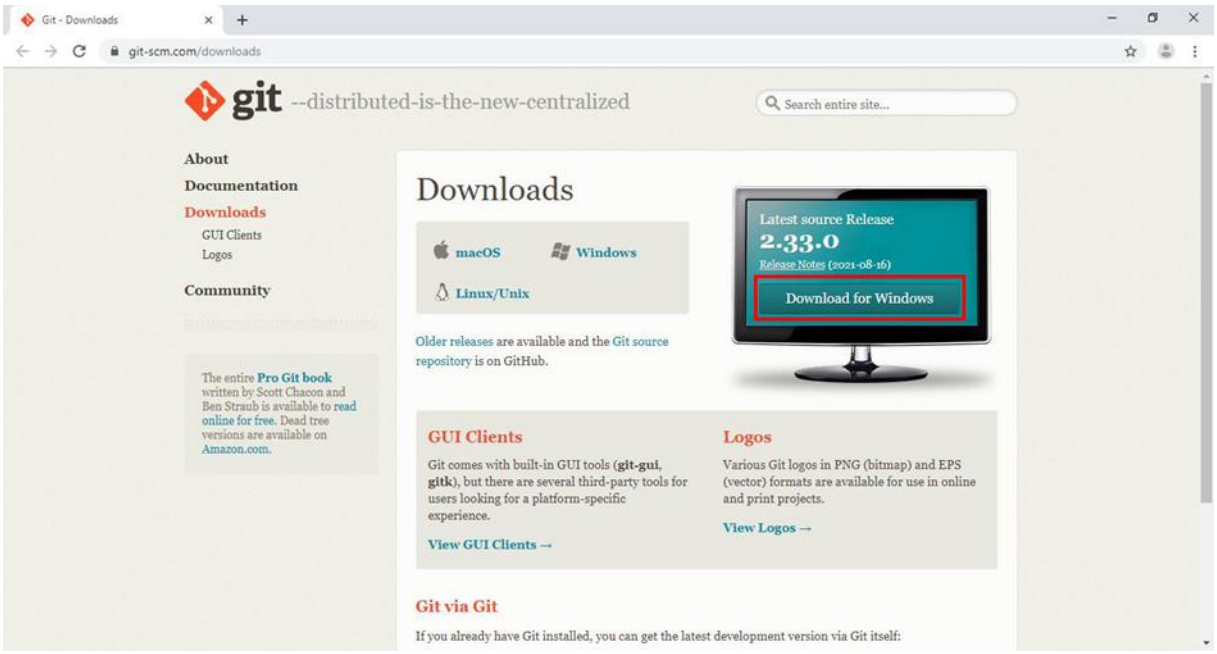
Installation sur macOS
Il existe plusieurs façons d'installer Git sur un Mac. Le plus simple est probablement d'installer les outils de ligne de commande Xcode. Sur Mavericks (10.9) ou supérieur, vous pouvez le faire simplement en essayant de taper la commande git à partir du terminal la toute première fois.

$ git --version

Si vous ne l'avez pas déjà installé, il vous demandera de l'installer.
Si vous voulez une version plus à jour, vous pouvez également l'installer via un programme d'installation binaire. Un programme d'installation macOS Git est maintenu et disponible en téléchargement sur le site Web de Git, à l'adresse https://git-scm.com/download/mac .

Installation Git Bash sur Windows

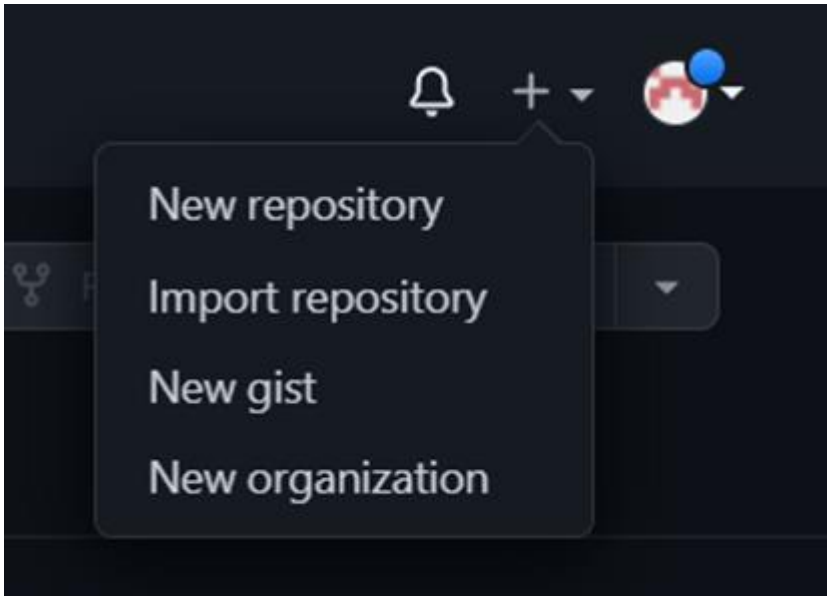
Télécharger le package Git pour Windows à partir du site Web du projet : https://git-scm.com/downloads



Maintenant que vous avez téléchargé l'exécutable Git Bash, vous allez exécuter le programme d'installation.

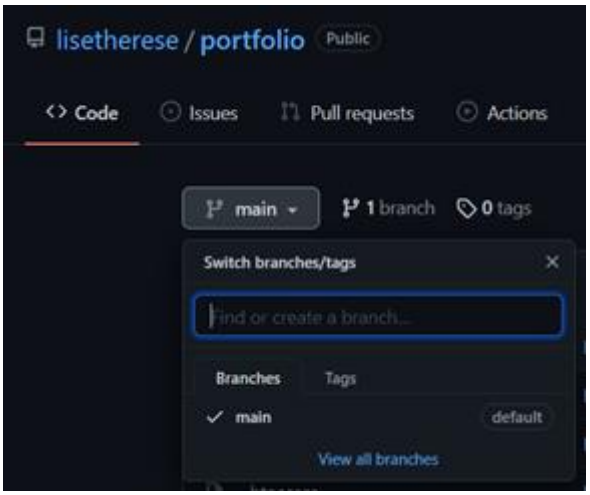
Exécuter le programme d'installation, installer le programme comme n'importe quelle autre application Windows, via le fichier .exe inclus.
Suivez les étapes sur le site Web https://www.stanleyulili.com/git/how-to-install-git-bash-on-windows/

Créer un dépôt
Pour faire quoi que ce soit dans GitHub, vous devez d'abord savoir comment démarrer un référentiel. Un référentiel (ou dépôt) est essentiellement synonyme du mot « projet ». Tout simplement, un référentiel stocke tout ce qui concerne un projet spécifique, notamment des fichiers, des images, des feuilles de calcul, des ensembles de données et des vidéos, souvent triés en fichiers. Il est préférable d'inclure un fichier README dans votre référentiel contenant des informations spécifiques concernant le projet donné. Sur GitHub, vous pouvez ajouter un fichier README dès que vous créez un nouveau référentiel.
Pour créer un nouveau référentiel, vous vous connecterez et cliquerez sur "New repository" dans le coin supérieur à droit. Vous pouvez ensuite nommer votre référentiel, inclure une brève description et cocher la case " Initialize this repository with a README". Enfin, vous cliquerez sur "Create repository".



Créer une branche

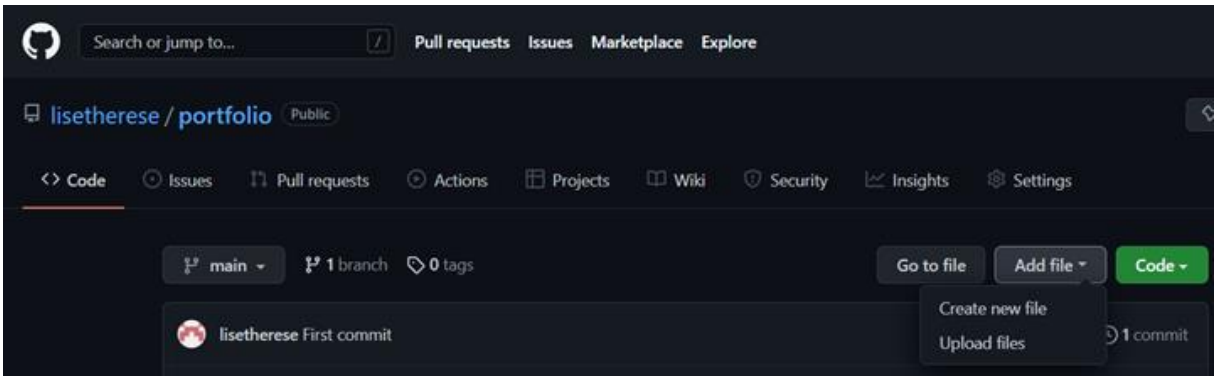
Les projets ont plusieurs facettes et de nombreuses versions de programme sont nécessaires lors de la construction. La création de branches vous permet de modifier simultanément plusieurs versions uniques d'un référentiel. Votre référentiel a automatiquement une branche définitive appelée master. Vous pouvez travailler sur plusieurs branches différentes afin d'apporter des modifications avant de les valider éventuellement dans la branche principale.
Lorsqu'une nouvelle branche est démarrée, ce sera une copie de la branche principale jusqu'à ce que vous la modifiiez pour apporter de nouvelles modifications. Une branche passe généralement par de nombreuses étapes et approbations avant d'être fusionnée dans la branche principale. Pour démarrer une nouvelle branche dans GitHub, accédez à votre nouveau référentiel, cliquez sur la liste déroulante qui lit "main", tapez un nom de branche, puis appuyez sur Entrée. Les branches sont idéales pour les nouvelles fonctionnalités ou les corrections de bogues.

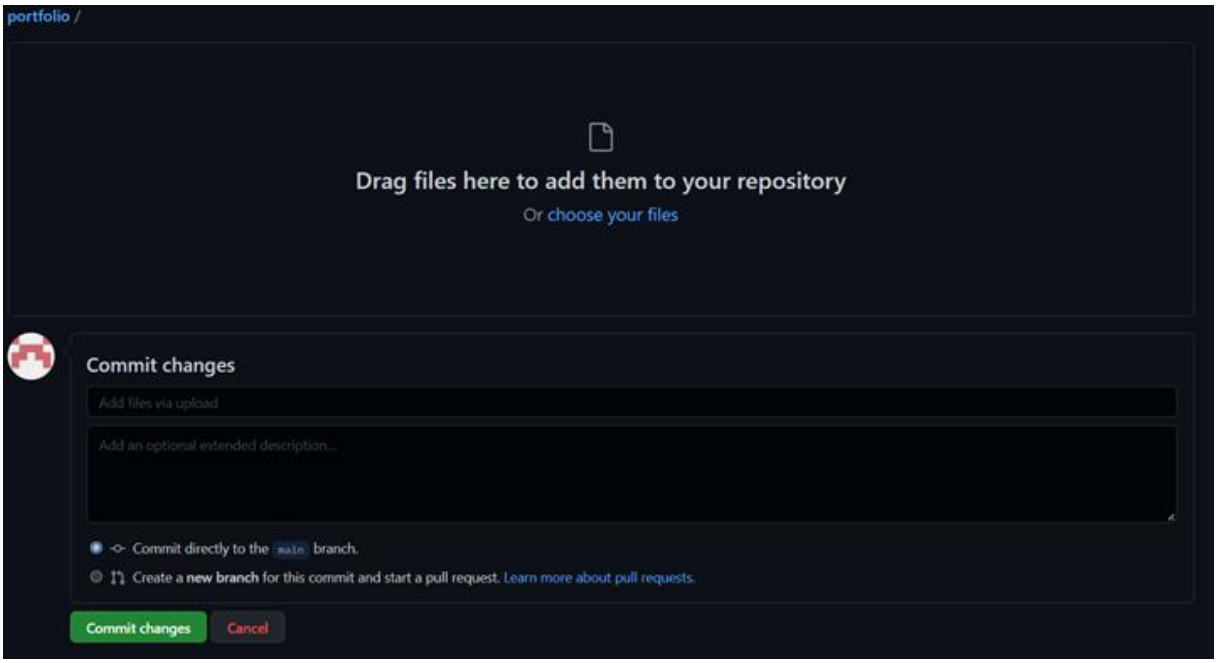


Créer et valider des modifications dans une branche

En utilisant la GitHub console
Pour créer ou télécharger de nouveaux fichiers et les ajouter à votre référentiel ou dépôt, accédez

simplement au bouton déroulant "Add file" et choisissez "Create new file" ou "Upload files", puis décrivez vos modifications en écrivant un message de validation, puis cliquez sur " commit changes ". Chaque modification enregistrée est appelée un commit. Chaque commit individuel a son propre message de commit qui donne plus de détails sur la raison pour laquelle un changement spécifique a été effectué. Les messages de validation donnent un historique des modifications et aident les contributeurs du projet à comprendre comment le projet a changé au fil du temps.

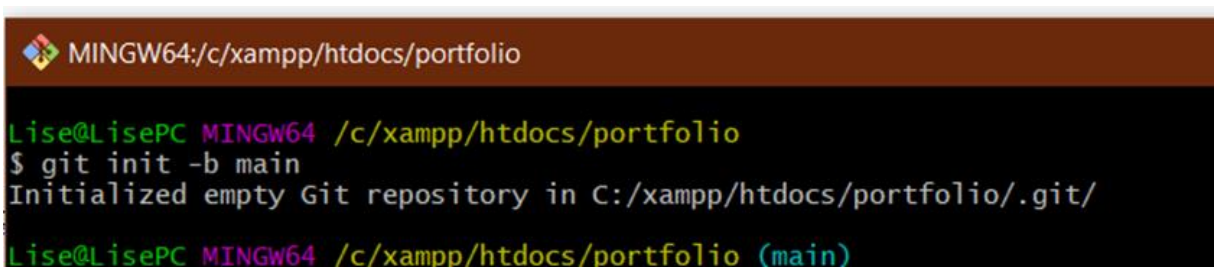




En utilisant Git ou Git Bash commandes

Accédez au dossier où vous voulez créer un dossier « .git » qui contient tous les fichiers que vous souhaitez télécharger sur votre dépôt sur Github, cliquez droit et puis choisissez « Git Bash ici » pour ouvrir Git Bash fenêtre de commande, ensuite tapez :

$ git init -b main

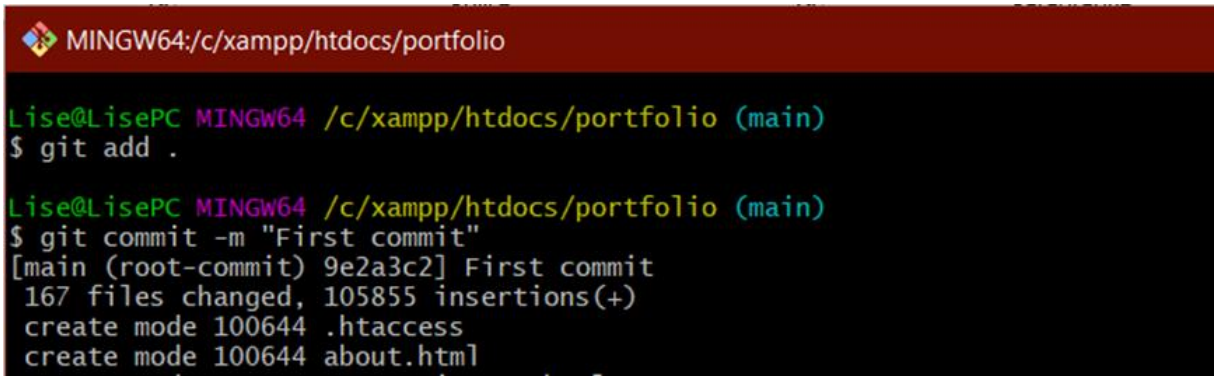


Ajoutez tous les fichiers existants dans ce dossier au dossier « .git » créé récemment par la commande

$ git add .

Faites ensuite le premier « commit » :

$ git commit -m « votre message commit ici »



Ou sinon vous pouvez grouper les deux commandes dans une ligne :

Créez votre dépôt sur GitHub (étape 3), copiez le lien qui se lie avec votre dépôt sur GitHub, puis tapez la commande sur Git Bash :

$ git remote add origin <le lien de votre dépôt>

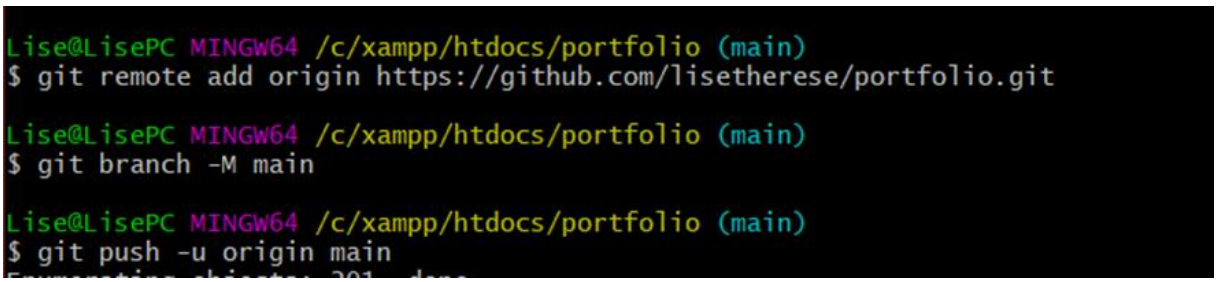
=> pour ajouter à distance un lien vers ce dossier Github à notre ".git" local. <le lien de votre dépôt> sur GitHub sera sous forme : https://github.com/votre-nom-d'utilisateur/votre-dépôt
=> dans ce tutoriel, on va utiliser : https://github.com/lisetherese/portfolio.git

$ git branch -m main

=> pour créer la nouvelle branche de votre projet pour ne pas nuire à nos originaux codes de projet. Ces nouvelles branches peuvent ensuite être utilisées pour tester les modifications apportées au code sans affecter le code principal du projet. Si les modifications fonctionnent, la branche peut être fusionnée avec la branche principale.

$ git push origin main

=> pour pousser vos modifications locales vers votre dépôt sur GitHub.

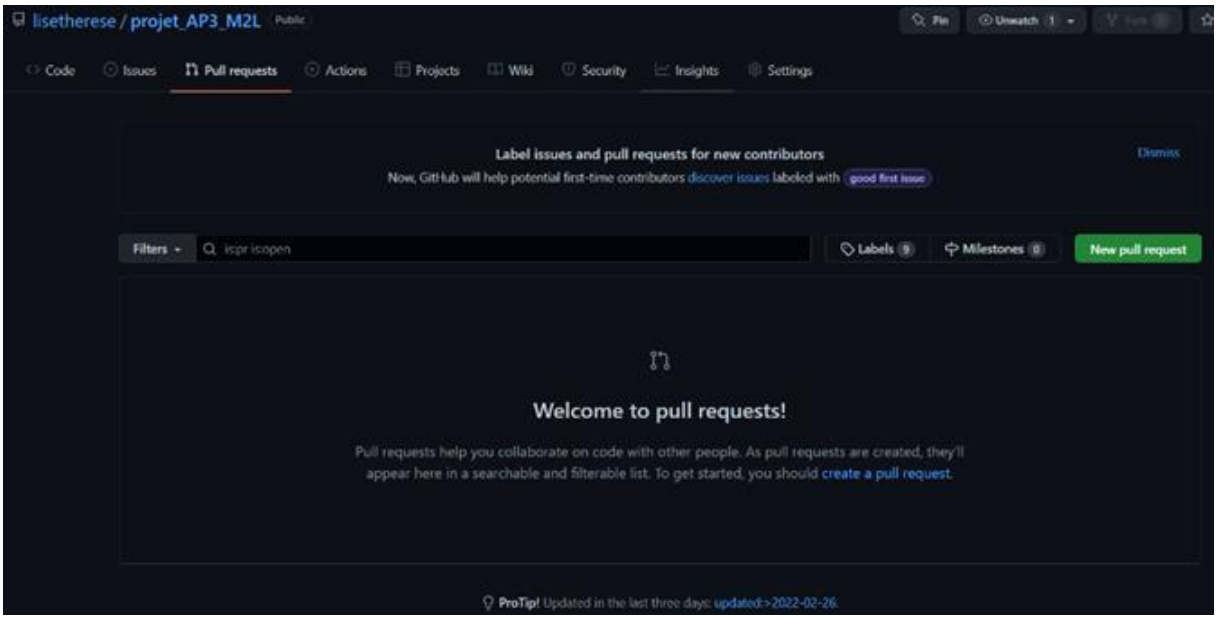


Demande d’extraction (pull request)

En utilisant la GitHub console

Pour qu'une branche soit fusionnée dans la branche d'une autre personne, vous devez ouvrir une demande d'extraction. Une demande d'extraction (pull request) est le moyen utilisé par GitHub pour informer les parties concernées de votre demande d'intégration des modifications dans’ leur branche. Une ‘pull request’ affichera en rouge et vert les différences de contenu entre les branches. Vous pouvez faire une ‘pull request’ chaque fois que vous terminez un commit. Pour de meilleurs résultats, lors de l'envoi d'une ‘pull request’, vous pouvez utiliser la fonction "@" pour mentionner des personnes spécifiques dont vous avez besoin de commentaires.

Pour ouvrir une demande d'extraction, vous allez dans l'onglet "Pull requests" et appuyez sur le bouton qui dit "New pull request". Ensuite, dans la case "Example comparisons" en bas, recherchez la branche que vous avez créée et comparez-la avec le ‘main’. Assurez-vous que vous aimez les modifications, puis cliquez sur le bouton "Create pull request". Intitulez votre ‘pull request’ et décrivez brièvement les changements. Pour terminer, cliquez sur "Create pull request".



En utilisant Git ou Git Bash commandes

Imaginez que vous construisiez votre travail sur votre branche ‘main’ et que vous vouliez qu'il soit intégré au projet. Vous devez d'abord envoyer cette modification à votre référentiel public pour que les autres puissent la voir :

$ git push https://github.com/lisetherese/portfolio.git main

Puis pour effectuer une git ‘pull request’ pour l'extraire vers votre référentiel local à partir de votre dépôt GitHub, tapez la commande :

$ git request-pull <le lien de votre dépôt sur GitHub>

Exemple

$ git request-pull https://github.com/lisetherese/portfolio.git main

Fusionnez la 'pull request'

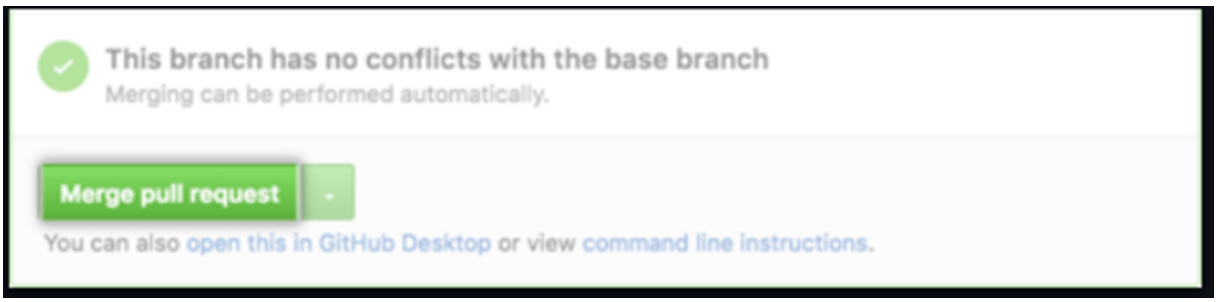
En utilisant la GitHub console

Dans votre dépôt sur GitHub, cliquez « Pull requests » (comme dans l'étape 6)

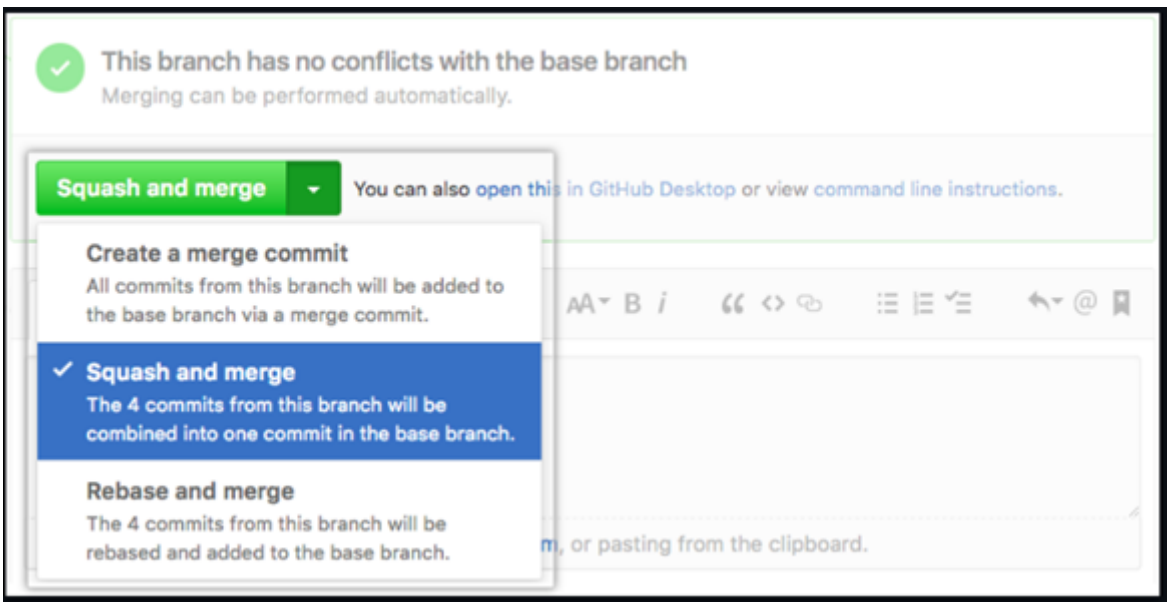
Dans la liste "Pull Requests", cliquez sur la 'pull request' que vous souhaitez fusionner.

En fonction des options de fusion activées pour votre dépôt (indiquées dans l'onglet « Settings » de votre dépôt, vous pouvez :

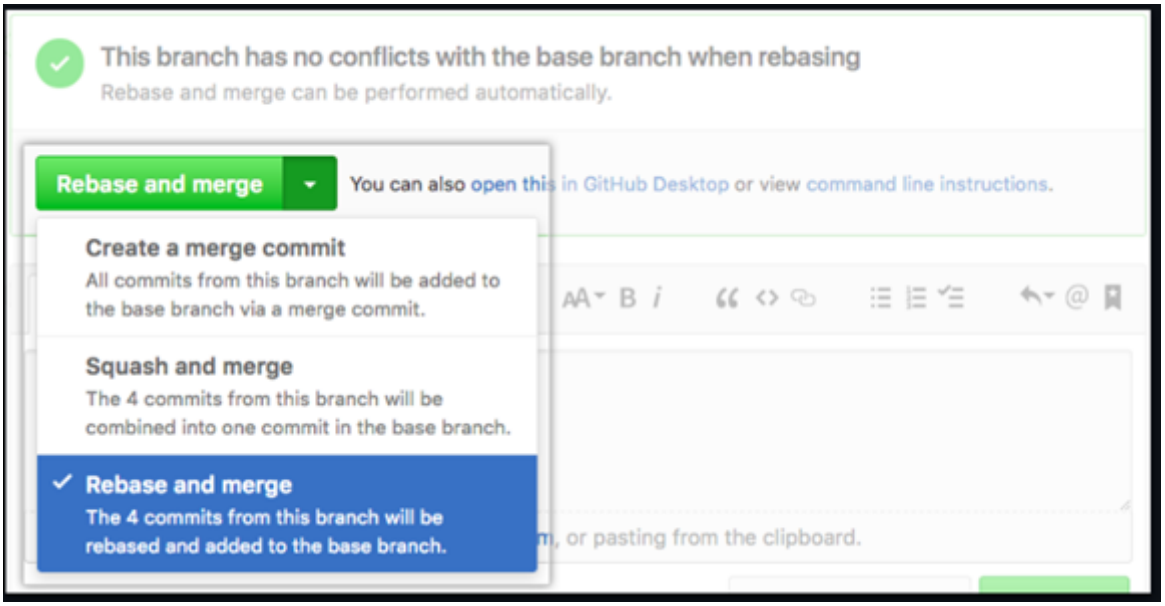
☑ « Merge pull request » : fusionnez tous les commits dans la branche de base en cliquant sur « Merge pull request ». Si l'option « Merge pull request » n'est pas affichée, cliquez sur le menu déroulant de fusion et sélectionnez « Create a merge commit ».



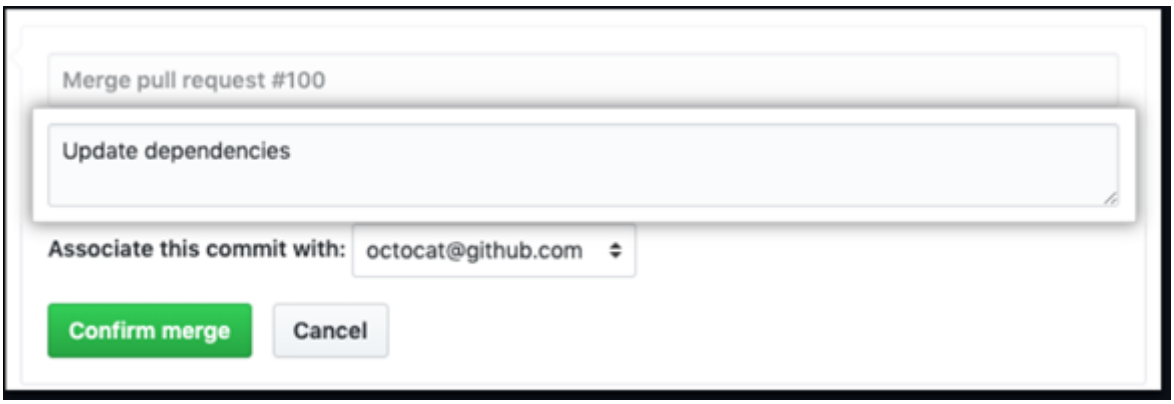
☑ Regroupez les commits en un seul commit en cliquant sur le menu déroulant de fusion, en sélectionnant « Squash and merge », puis en cliquant sur le bouton « Squash and merge».



☑ Rebasez les commits individuellement sur la branche de base en cliquant sur le menu déroulant de fusion, en sélectionnant « Rebase and merge », puis en cliquant sur le bouton « Rebase and merge ».

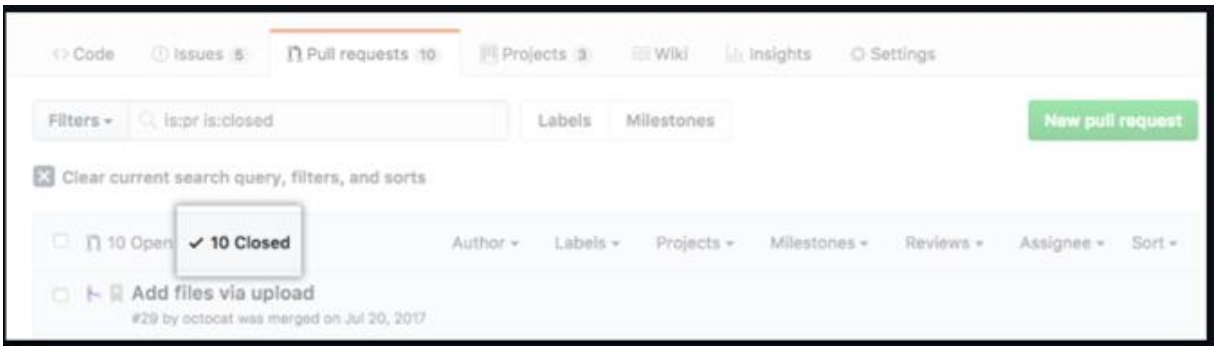


☑ Si vous y êtes invité, saisissez un message de validation ou acceptez le message par défaut. Puis cliquez le button « Confirm merge »



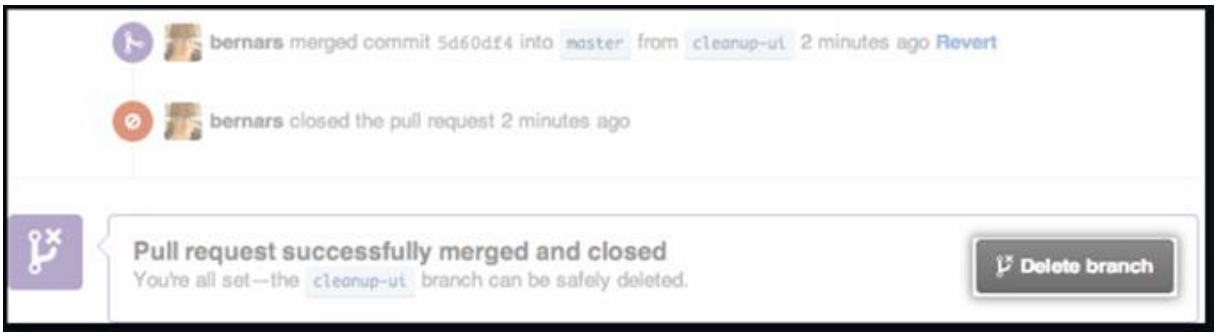
Ensuite, supprimez la branche que vous avez fusionnée une fois qu'elle a été incorporée dans le ‘main’ pour garder la liste des branches de votre dépôt bien rangée.

☑ Dans votre onglet « Pull requests » (comme dans l’étape 6), cliquez sur « Closed » pour afficher une liste des demandes d'extraction fermées.



☑ Dans cette liste, cliquez sur la ‘pull request ‘ associée à la branche que vous souhaitez supprimer.

☑ Au bas de cette ‘pull request’, cliquez sur ‘Delete branch ‘.



En utilisant Git ou Git Bash commandes

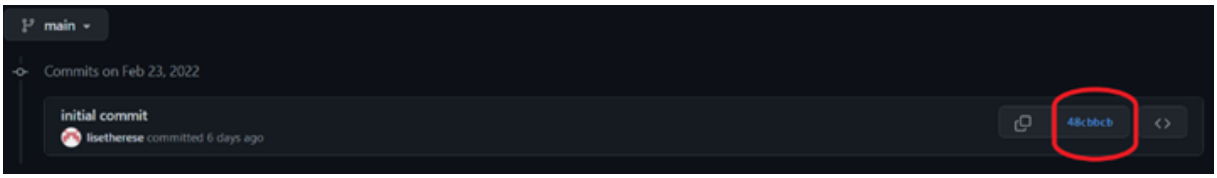
Les lignes de commande correspondantes aux différents types de fusionner :

☑ Le bouton "Merge pull request" sur GitHub correspond à :

$ git checkout <-branch>

$ git merge --no-ff -m <-message> <-commit-hash>

=> La partie <-branch> est votre branche cible : celle sur laquelle vous voulez que le commit de fusion soit, une fois la fusion terminée.
=> La partie est quelque chose que vous devriez générer à la main.
=> La partie est obligatoire à ce stade, il s'agit du code de hachage de l'objet de validation dans votre dépôt qui peut être trouvé facilement dans la console GitHub
Par cliquez sur le commit ciblé.



☑ Le bouton « Squash and merge” sur GitHub console correspond à :
$ git checkout <-branch>

$ git merge --squash <-commit-hash>

$ git commit

☑ Le bouton “Rebase and merge » sur GitHub console correspond à :
$ git checkout <-commit-hash-or-branch-name>

$ git rebase <-branch>

$ git checkout <-branch>

$ git merge --ff-only <-hash-or-name>

Ensuite, supprimez la branche que vous avez fusionnée une fois qu'elle a été incorporée dans le ‘main’ :

☑ Tout d'abord, nous affichons toutes les branches (locales et distantes), en utilisant la commande ‘git branch’ avec l'indicateur -a (all).

$ git branch -a

☑ Pour supprimer la branche locale, il suffit d'exécuter à nouveau la commande ‘git branch’, cette fois avec l’indicateur -d (delete), suivi du nom de la branche que vous souhaitez supprimer (branche de test dans ce cas).

$ git branch -d <-branch>

```
git branch -a
# *master
#  test
#  remote/origin/master
#  remote/origin/test

git branch -d test
# Deleted branch test (was ########).
```

☑ Pour supprimer la branche à distant sur GitHub, vous ne pouvez pas utiliser la commande ‘git branch’. À la place, utilisez la commande ‘git push’ avec l'indicateur --delete, suivi du nom de la branche que vous souhaitez supprimer. Vous devez également spécifier le nom distant (origine dans ce cas) après git push.

$ git push origin –delete <-branch>

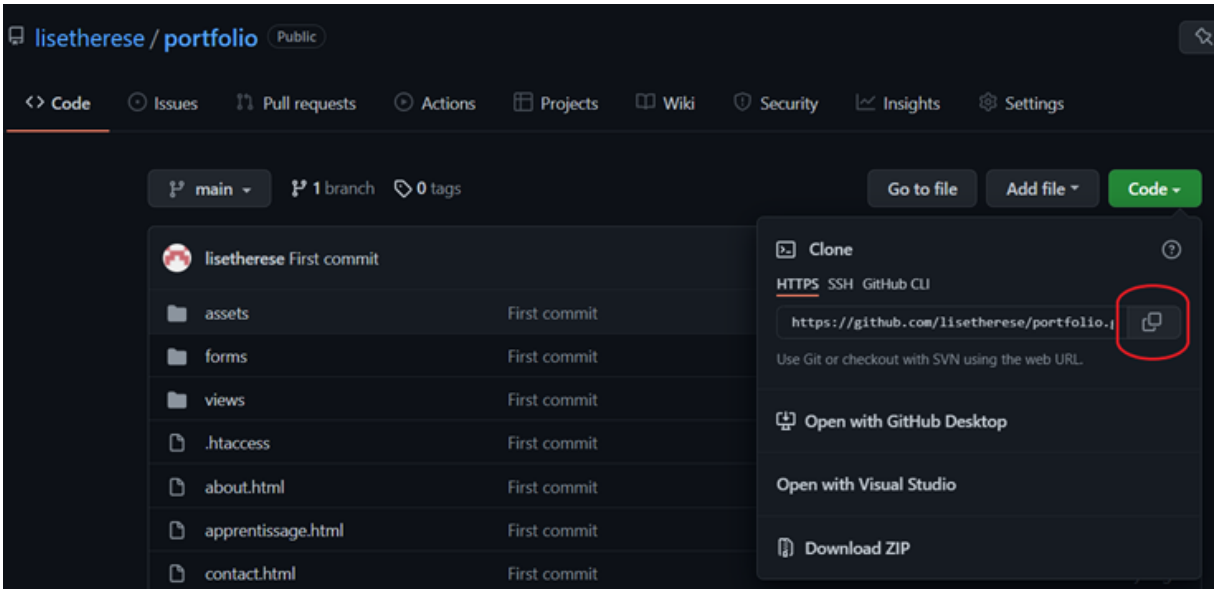
```
git branch -a
# *master
#  test
#  remote/origin/master
#  remote/origin/test

git push origin --delete test
# To <URL of your repository>.git
#  - [deleted]         test
```

D. Cloner un dépôt (clone)

Vous pouvez cloner un dépôt de GitHub sur votre ordinateur local pour faciliter la résolution des conflits de fusion, ajouter ou supprimer des fichiers et pousser des validations plus importantes. Lorsque vous clonez un dépôt, vous le copiez vers votre ordinateur local.
Le clonage d'un dépôt extrait une copie complète de toutes les données du dépôt que GitHub possède à ce moment-là, y compris toutes les versions de chaque fichier et dossier du projet.

Sur GitHub, accédez à la page principale du dépôt, cliquez sur le bouton déroulant "Code". Pour cloner le dépôt en HTTPS, sous "Clone with HTTPS", cliquez sur l'icône comme dans la photo :



Ouvrez Git Bash, changez le répertoire de travail actuel à l'emplacement où vous voulez que le répertoire cloné, tapez la commande ‘git clone’ pour faire le clonage vers votre machine locale :

$ git clone <le lien de votre dépôt sur GitHub>

Exemple

$ git clone https://github.com/lisetherese/portfolio.git