

Introduction au dépôt distant avec Gitlab

Attribution - Partage dans les Mêmes Conditions : <http://creativecommons.org/licenses/by-sa/4.0/fr/>

Table des matières

1. Créer un dépôt distant	3
2. Cloner un dépôt distant	5
3. Tirer des changements	6
4. Pousser des commits	7
5. Collaborer avec Gitlab : droits, issues, commentaires	8
6. Gérer la concurrence et les conflits	10
7. Défi : Veille	12
Solutions des exercices	14

1. Créer un dépôt distant

Remote

Az Définition

Une *remote repository* (ou dépôt distant) est un dépôt Git en ligne.
Il se comporte comme un *repository* local, il stocke donc un ensemble de commits.

Intérêt des remotes

💡 Fondamental

- Partage du dépôt git : on peut désormais travailler à plusieurs dessus
- Sauvegarde du travail à distance

Forge

Az Définition

Une forge est un système de gestion de versions associé à des fonctions collaboratives (discussions, gestions de bugs...).

Une forge Git est une forme qui repose sur Git comme système de gestion de version. Les forges Git sont aujourd'hui parmi les plus répandues.

Github

👁 Exemple

Github est une énorme forge Git centralisée propriété de Microsoft depuis 2018.
github.com

Gitlab

👁 Exemple

Gitlab est un logiciel permettant d'installer sa propre forge Git dans une logique décentralisée. Framasoft et l'UTC hébergent chacune une instance de la forge Gitlab (il en existe beaucoup d'autres).
gitlab.utc.fr
framagit.org

Création d'un projet (dépôt distant)

 Méthode

Pour le Gitlab de l'UTC :

- Se rendre sur <https://gitlab.utc.fr>
- Une fois connecté avec ses identifiants CAS, il suffit d'appuyer sur le bouton « New project » et de renseigner un nom et un niveau de visibilité

Ajout de clé SSH

 Complément

Pour simplifier l'authentification au serveur, on peut ajouter sa clé publique SSH à son profil.

2. Cloner un dépôt distant

Cloner un dépôt

 Méthode

Cloner un dépôt Git, c'est récupérer une copie du *remote repository* sur sa machine. Cela fera en même temps la configuration nécessaire pour que :

- un dépôt local soit créé,
- le dépôt distant soit lié au dépôt local.

 Syntaxe

À partir du dossier dans lequel on souhaite créer le dépôt local :

```
1 git clone url .
```

 Exemple

```
1 mkdir helloworld
2 cd helloworld
3 git clone https://framagit.org/stph.crzt/helloworld .
```

3. Tirer des changements

Méthode

Tirer des changements, c'est récupérer des commits depuis le dépôt distant.

Syntaxe

```
1 git pull
```

Cette commande ajoute les commits distants au dépôt local.

git pull et working directory

Attention

Si la «copie de travail» n'est pas propre, c'est à dire contenant des modifications non-référencées, et que les fichiers concernés existent sur le dépôt distant, alors `git pull` affichera un message d'erreur vous invitant à valider ou annuler vos modifications locales. Cela ne concerne pas les fichiers existant uniquement dans l'espace de travail.

git pull et dépôt local

Attention

Si votre dépôt local et le dépôt distant ont divergés et au moins un fichier a été modifié de chaque côté, alors `git pull` créera automatiquement un commit de fusion. Si une même ligne dans un même fichier a été modifiée des deux côté, alors il y a un conflit.

Fusion et conflits

Remarque

Parfois, bien qu'il y ait de nouveaux commits sur le dépôt distant, `git pull` indique que le dépôt local est à jour. Pour forcer `git` à télécharger les nouvelles références sur le dépôt distant, on peut utiliser la commande `git fetch`.

4. Pousser des commits

Méthode

Pousser un ou plusieurs commits, c'est envoyer du travail local sur le serveur pour le sauvegarder et éventuellement le partager.

Syntaxe

```
1 git push
```

Cette commande ajoute les commits locaux au dépôt distant.

Attention

Par défaut, `git push` retournera une erreur si votre dépôt local a divergé par rapport au dépôt distant, pour éviter d'écraser par erreur les commits côté serveur.

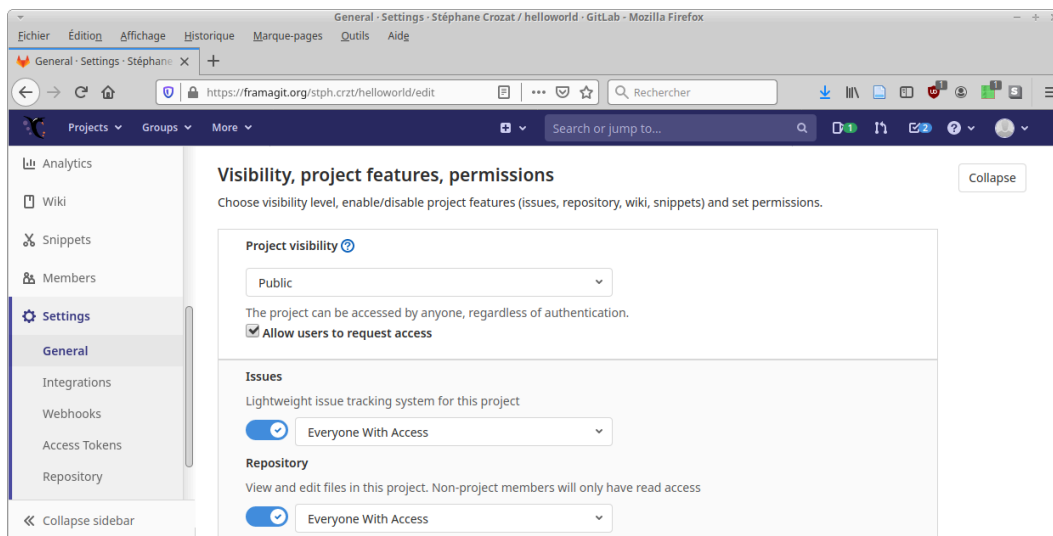
Il faut donc penser à faire un `git pull` avant un `git push`.

5. Collaborer avec Gitlab : droits, issues, commentaires

Gérer les droits

Méthode

- Visibilité du projet (droit de le cloner)
- Droits sur le projet (droit de pousser des modifications)
- Droits sur les issues (droit de proposer des issues)



<https://framagit.org/stph.crzt/helloworld/edit>

Issue

Az Définition

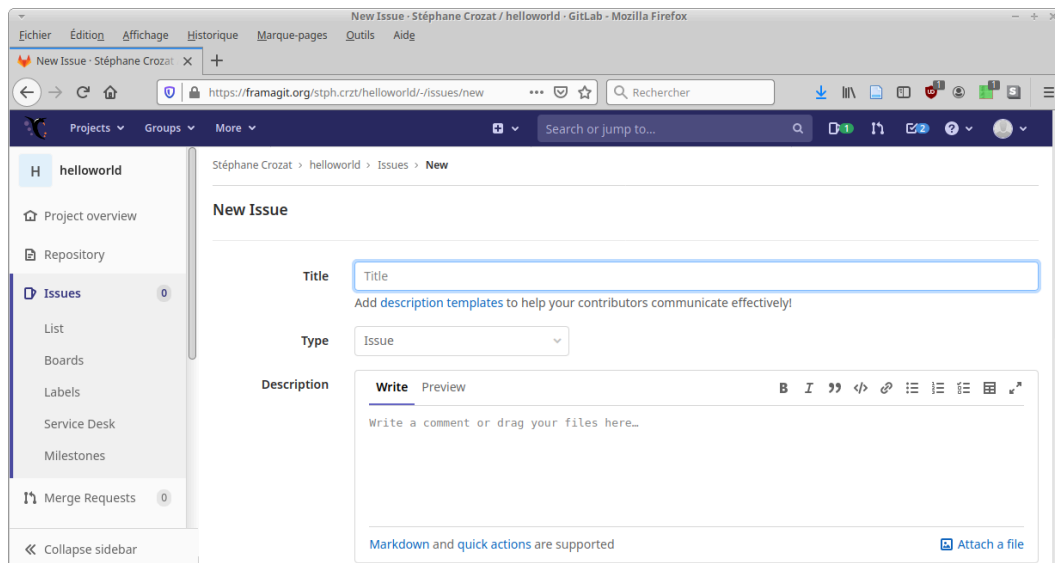
Une issue est le relevé d'un problème ou d'une demande d'amélioration effectué sur la forge

Il y a plusieurs intérêts, par exemple :

- Cela facilite la communication entre les différents acteurs (on peut s'en servir pour faire remonter les erreurs rencontrées).
- Cela permet d'assurer un suivi (on peut lister toutes les tâches à réaliser sous la forme d'issues).
- Cela permet d'organiser le travail à plusieurs (on peut assigner une issue à quelqu'un).
- Cela permet de discuter sur les évolutions (chaque issue peut servir d'espace de discussion contextuel).
- On peut référencer les issues dans les messages de commit (souvent pour signifier que l'on a résolu cette issue).

Gérer les évolutions avec les issues

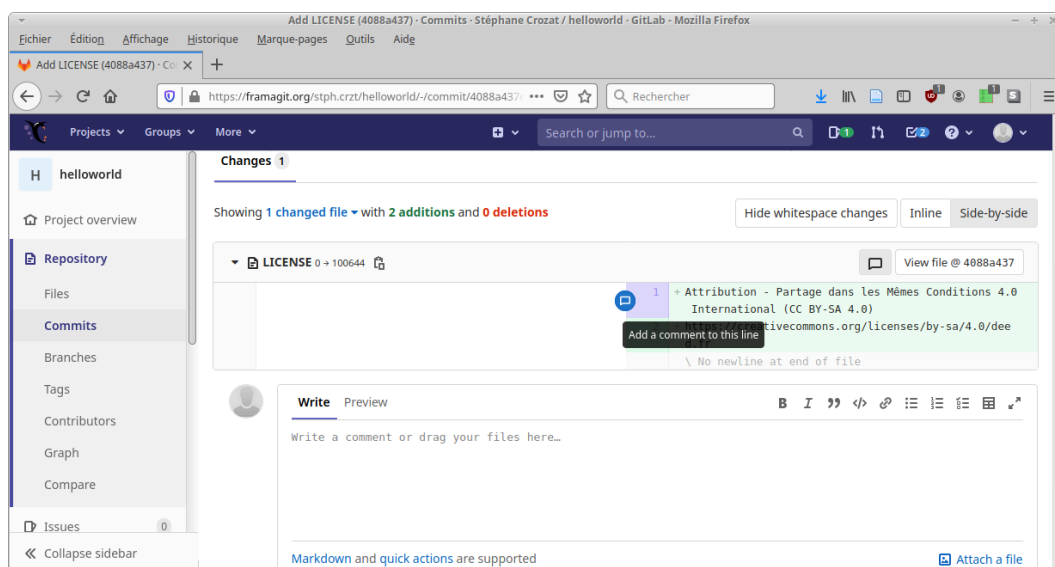
 Méthode




<https://framagit.org/stph.crzt/helloworld/-/commit/ed2374656ae75dee02c9ae72a5b93abcbfc5482f>

Commenter les commits

 Méthode



<https://framagit.org/stph.crzt/helloworld/-/issues/new>

 Remarque

Il est possible de commenter chaque ligne des *diff*.

6. Gérer la concurrence et les conflits

⚠ Attention

Si deux personnes sont autorisées à pousser sur le même dépôt alors il y a un risque de conflit.

Un conflit survient quand deux dépôt locaux ont produit des modifications localement concernant les mêmes fichiers et cherchent ensuite à les pousser sur le même dépôt distant.

💬 Remarque

Si une seule personne utilise deux dépôt locaux connectés au même dépôt distant (par exemple pour travailler sur deux machines différentes), alors il y a aussi risque de conflit.

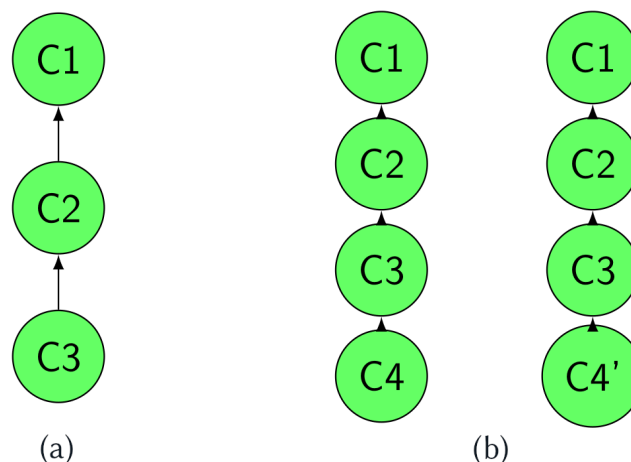
Conflit

Az Définition

On parle de conflit lorsque deux personnes ont modifié les mêmes lignes d'un fichier en parallèle et que Git ne peut donc pas savoir quelle version conserver lors d'une fusion.

- Git ne comprend pas le texte qu'il versionne, il ne sait donc pas quoi garder en cas de conflit.
- Résoudre un conflit c'est dire à Git quelle est la bonne version à garder.
- Il peut s'agir d'une des deux versions uniquement, ou d'un mélange des deux.

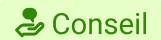
👁 Exemple



- Alice tire le projet en (a), il y a 3 commits.
- Alice ajoute un commit C4 sur son dépôt local.
- Pendant ce temps, Bob ajoute son commit C4' sur le projet et **le pousse sur le dépôt distant**.

- Si Alice veut pousser sa modification C4 alors elle doit d'abord tirer les modifications de Bob et en cas de conflit choisir quelles modifications parmi celles de Bob sont pertinentes.

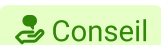
Bonnes pratiques



La résolution des conflits n'est pas toujours une tâche aisée, surtout pour des débutants, il est conseillé de :

- éviter les conflits en se coordonnant (savoir qui fait quoi et quand sur le dépôt distant),
- lorsqu'un conflit survient, bien prendre le temps de le résoudre en consultant attentivement les fichiers concernés.

Conseil de brute



Si :

1. un conflit survient,
2. que vous avez des modifications auxquelles vous tenez en local,
3. que vous n'êtes pas sûr de ce que vous allez faire parce que vous ne connaissez pas bien Git,

alors :

- je vous conseille de copier le contenu de votre dossier "à l'ancienne" (dans un dossier temporaire), avant de résoudre votre conflit.



La bonne gestion des conflits suppose (notamment) l'usage des fonctions Git de type *branch* et *merge* et l'usage des fonctions Gitlab de type *merge request*.

7. Défi : Veille

Question 1

[solution n°1 p. 14]

Créez un dépôt distant (projet) sur le Gitlab de l'UTC que vous intitulerez Veille :

- créez un fichier README.md avec votre nom (ou un pseudo),
- associez une licence (compatible avec celle de Wikipédia),
- clonez le projet en local.

Question 2

[solution n°2 p. 14]

Ajoutez un fichier `prism.md` qui contient le premier paragraphe concernant ce programme de la NSA sur Wikipédia.

Commitez et poussez ce fichier.

Indice :

fr.wikipedia.org/wiki/PRISM¹

Question 3

[solution n°3 p. 14]

Modifiez votre fichier afin d'ajouter la fin de l'introduction présente sur Wikipédia, commitez et poussez.

Question 4

[solution n°4 p. 14]

Supprimez votre dépôt local puis créez à nouveau le dépôt.

Accordez-vous avec un autre utilisateur que nous appellerons Édouard dans le cadre de cet exercice.

Question 5

[solution n°5 p. 14]

Autorisez Édouard à pousser sur votre dépôt (et demandez-lui de faire de même pour vous).

Question 6

[solution n°6 p. 14]

Clonez le dépôt d'Édouard et poussez un fichier `chelsea.md` sur son dépôt qui contient le premier paragraphe de présentation de Chelsea Manning sur Wikipédia.

Indice :

fr.wikipedia.org/wiki/Chelsea_Manning

Question 7

[solution n°7 p. 14]

Commentez le premier commit d'Édouard afin de lui conseiller d'ajouter un email à son README.

¹ [https://fr.wikipedia.org/wiki/PRISM_\(programme_de_surveillance\)](https://fr.wikipedia.org/wiki/PRISM_(programme_de_surveillance))

Question 8

[solution n°8 p. 14]

Ajoutez une issue à votre projet afin de prévoir l'ajout d'un document sur Aaron Swartz.

Indice :

fr.wikipedia.org/wiki/Aaron_Swartz

Question 9

[solution n°9 p. 14]

Ajoutez une issue au projet d'Édouard afin de lui demander l'ajout d'un document sur Wikileaks.

Indice :

fr.wikipedia.org/wiki/WikiLeaks

Solutions des exercices

Solution n°1

[exercice p. 12]

Solution n°2

[exercice p. 12]

Solution n°3

[exercice p. 12]

Solution n°4

[exercice p. 12]

Solution n°5

[exercice p. 12]

Solution n°6

[exercice p. 12]

Solution n°7

[exercice p. 12]

Solution n°8

[exercice p. 13]

Solution n°9

[exercice p. 13]