Tutoriel GIT – BlueJ – Projet Zuul v2.1

GIT

“Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.”
Et pour les plus curieux qui veulent savoir d’où vient le nom GIT (prononcez « guitte ») :
https://en.wikipedia.org/wiki/Git#Naming

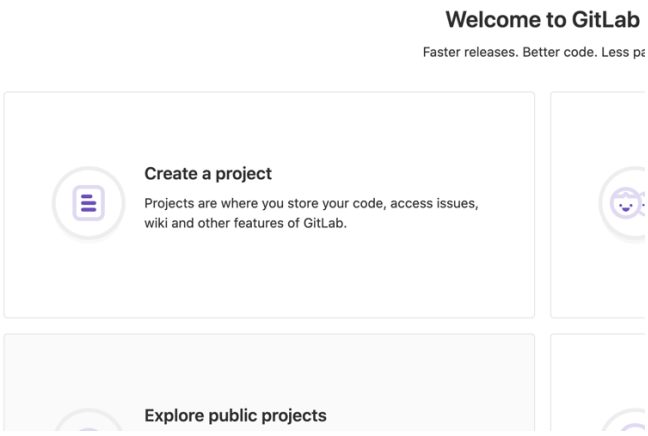
Git est devenu le standard de la gestion de projets informatiques aussi bien en entreprise, que pour des projets open source. Cet outil indispensable permet de travailler efficacement en équipe sur un même projet et, même lorsqu’on est seul développeur, de créer un historique des modifications apportées au projet. Autrement dit il permet un contrôle des versions du projet, avec la possibilité de revenir en arrière.

Création d’un projet GitLab

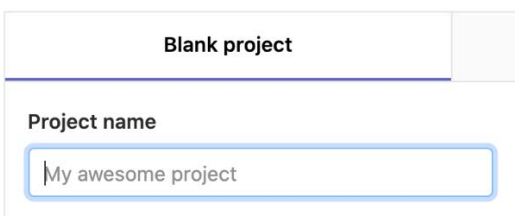
Tout d’abord il nous faut créer ce qu’on appelle un projet sur le site GitLab. GitLab est un serveur hébergé dédié à la gestion de projets via l’outil git.

À noter la différence entre GIT et GitLab : Git est l’outil utilisé, et GitLab un site permettant d’héberger les versions et historique du projet. De nombreux autres sites le permettent aussi (GitHub est le plus connu par exemple). Nous allons utiliser une version locale à l’ESIEE : https://git.esiee.fr/

Après t’être identifié, il faut cliquer sur *create a project*

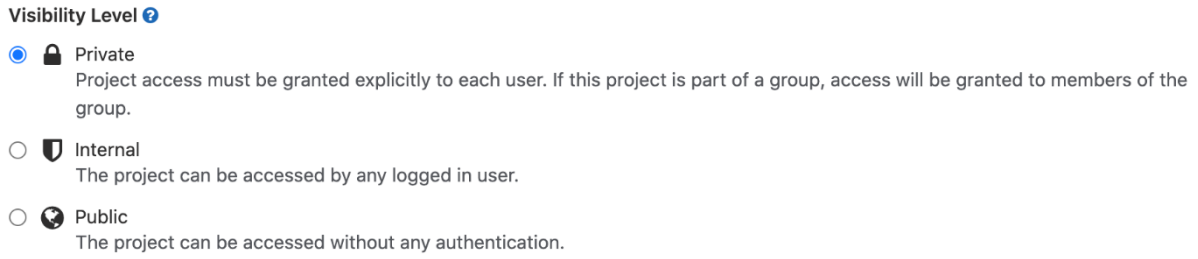


Donne un nom à ton projet :



Tu peux également lui donner une description (à noter qu'une description pourra être ajoutée plus tard dans les réglages du projet, une fois celui-ci créé).

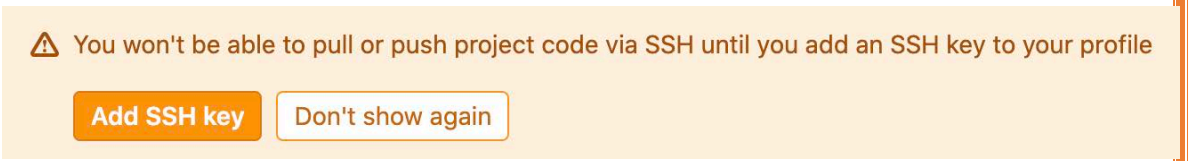
La visibilité de ton projet devrait rester privée.



Étant donné que nous allons « pusher » (pousser, ou encore « upstreamer ») notre projet Zuul de notre machine vers le serveur de GitLab, **ne** coche **pas** la case suivante. Inutile de laisser GitLab ajouter un fichier Readme.md, on le fera à la main plus tard.

Note :

Si tu n'as pas de clé SSH, un message te le signalera. Cette clé sert à certifier les échanges entre ta machine et les serveurs de GitLab (push, pull, …).
Ce n'est pas nécessaire pour notre utilisation avec BlueJ. Nous allons passer en HTTPS.
Ne tiens donc pas compte du message suivant :

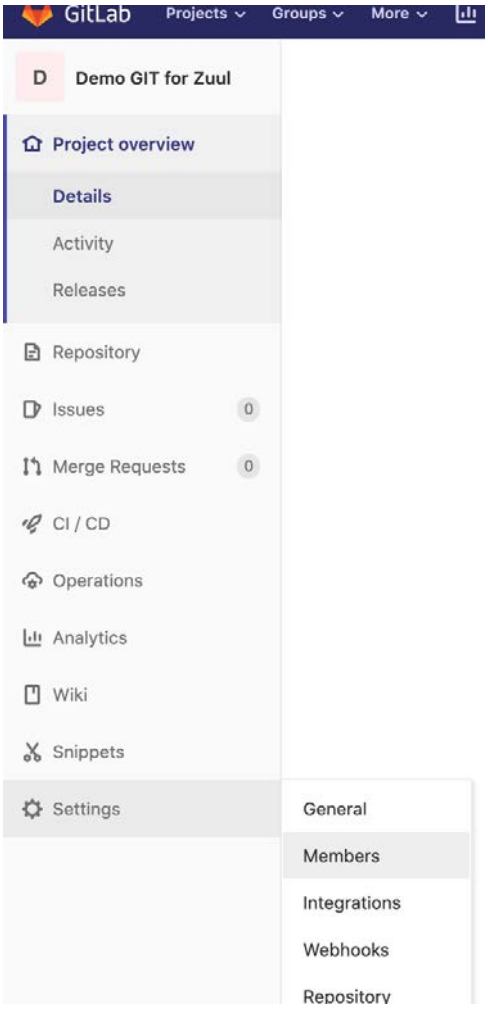


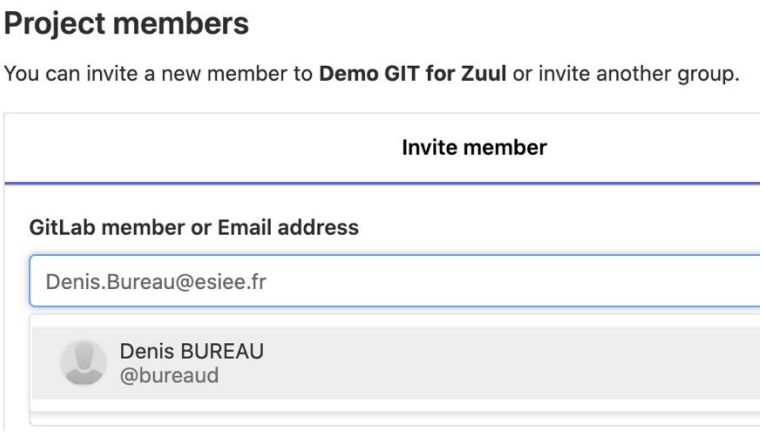
Ton projet GitLab est maintenant créé et vide.

Partage de ton projet

Obligatoire :

Dans les **Settings** de ton projet sur GitLab, clique sur **Members**.
Puis ajoute Mr Denis Bureau indiqué par **@bureaud**, et choisis le rôle **Maintainer** **(**normalement, l'adresse Denis.Bureau@esiee.fr est automatiquement connue de GitLab).
Tu devras peut-être ajouter d'autres intervenants plus tard.
La démarche à suivre sera la même, leur adresse mail te sera communiquée.

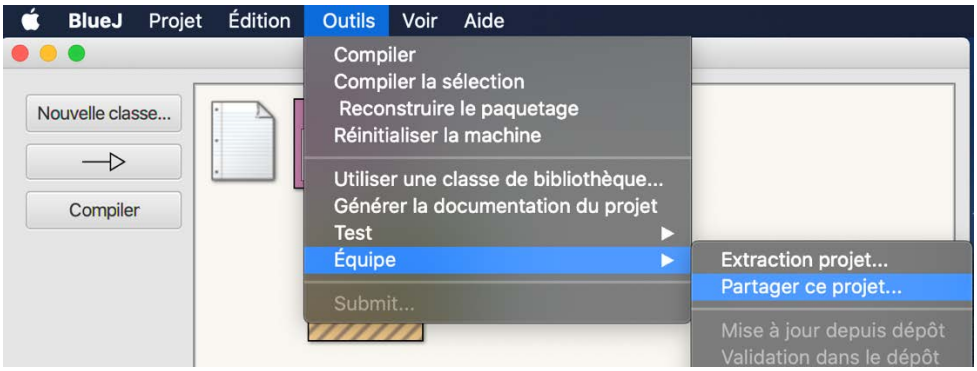




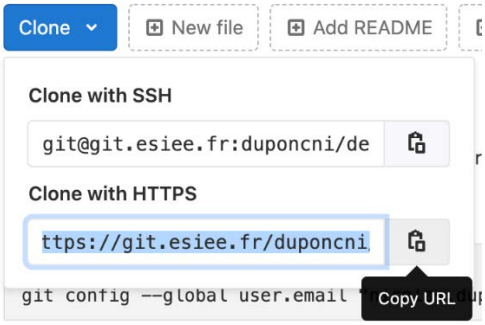
Synchronisation du projet local vers GitLab

BlueJ dispose d'une interface graphique permettant de ne pas avoir à configurer git via le terminal (en ligne de commande).

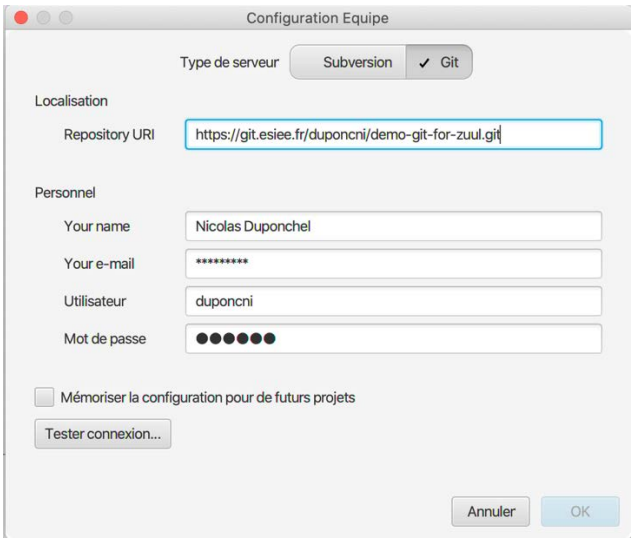
Partage du projet :



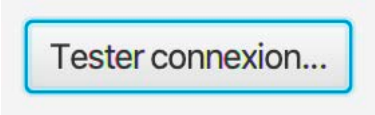
Voici où tu peux trouver l'URL https de ton projet GitLab :
Pour récupérer l'URL HTTPS de ton projet, il faut aller sur GitLab, selectionner ton projet. Une fois sur la page principale, **Clique sur le bouton clone** en bleu pour voir apparaitre les liens (selon l'interface, il faut cliquer sur la petite flèche), et choisis « clone with HTTPS » :



Remplis les informations nécessaires à la configuration de git :



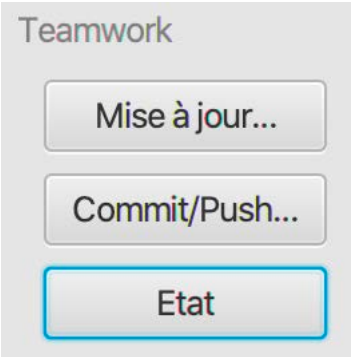
La bonne pratique : teste la connexion avant de valider.



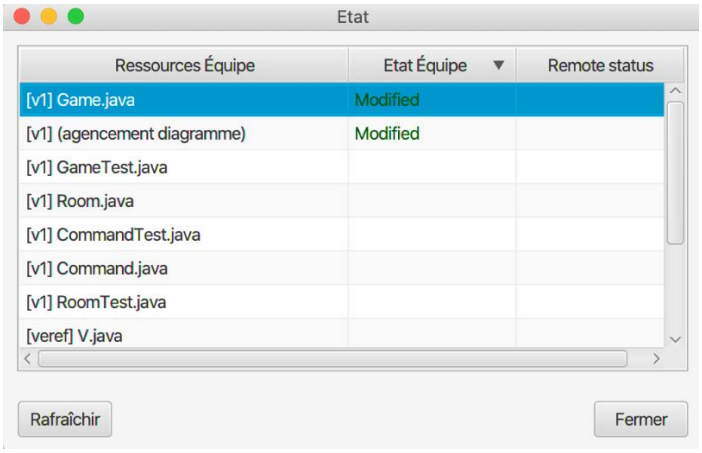
Ton projet Zuul local est maintenant poussé vers ton repo GitLab. Rafraichis la page de ton projet sur GitLab, tu pourras accéder aux dossiers et fichiers de ton projet.
Tu es maintenant prêt à travailler avec git tout au long du projet Zuul, avec une sécurité inégalée en cas de crash de ton disque dur, de vol de ton ordinateur portable, ou d'oubli de la dernière version à l'école ou à la maison.

État du projet

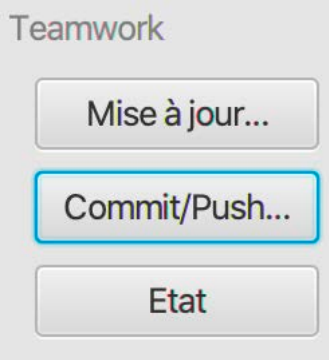
Le bouton Etat dans BlueJ permet d'accéder à une fenêtre dans laquelle sont listés les fichiers qui ont subi des modifications depuis la dernière synchronisation du projet.



Par exemple, je viens de modifier la classe Game. Si je regarde l'état du projet, je constate que la classe Game a bien été modifiée.



Synchronisation des modifications

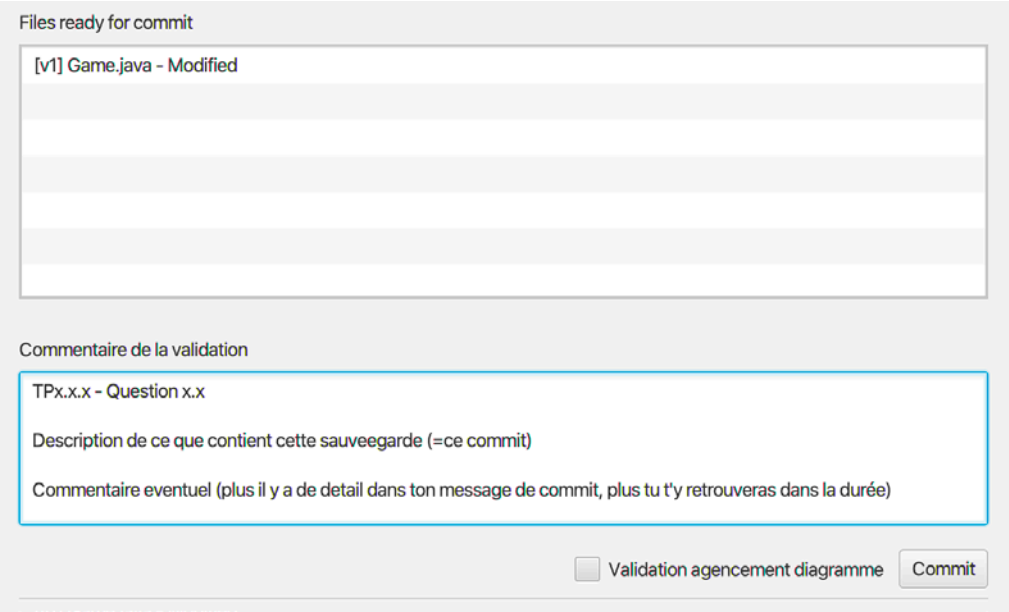


Pour synchroniser tes modifications, git te permet de faire les choses suivantes :

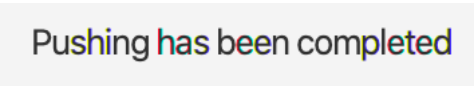
- « Commiter » : créer une sauvegarde de tes modifications en cours.
- « Pusher » : envoyer les commits contenant les modifications sur GitLab.

BlueJ dispose d'une fonction intégrée qui fait tout ça automatiquement. Clique sur Commit/Push.

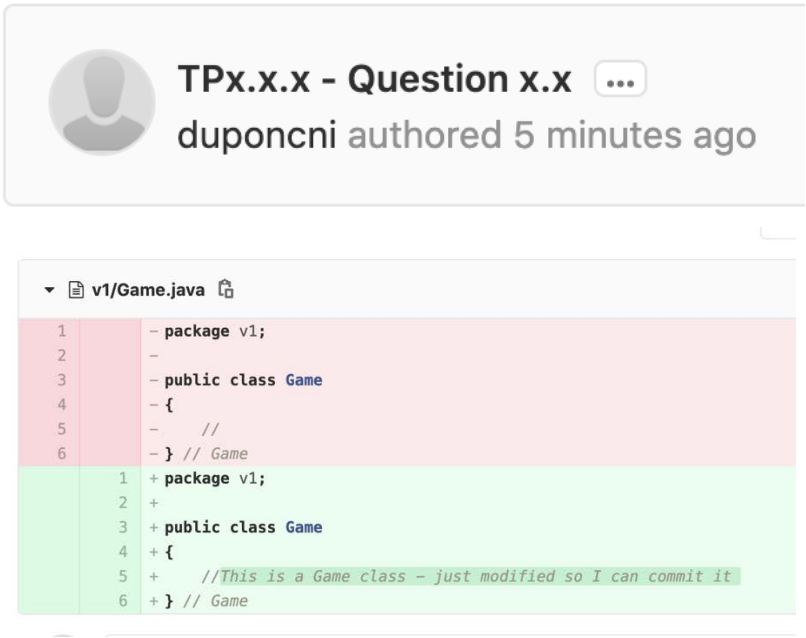
Tu peux voir les fichiers contenant des modifications, ou les nouveaux fichiers ajoutés au projet, dans « Files ready for commit ». Ajoute une description à ton commit, sois le plus précis possible, tu verras que c'est important dans la durée du projet pour t'y retrouver plus facilement (par exemple le numéro d'exercice). Tu peux maintenant « commiter » :



Maintenant tu vas pouvoir « pusher ton commit » vers le repo hébergé sur GitLab. A noter que tu peux faire plusieurs commits avant de « pusher » ton travail.

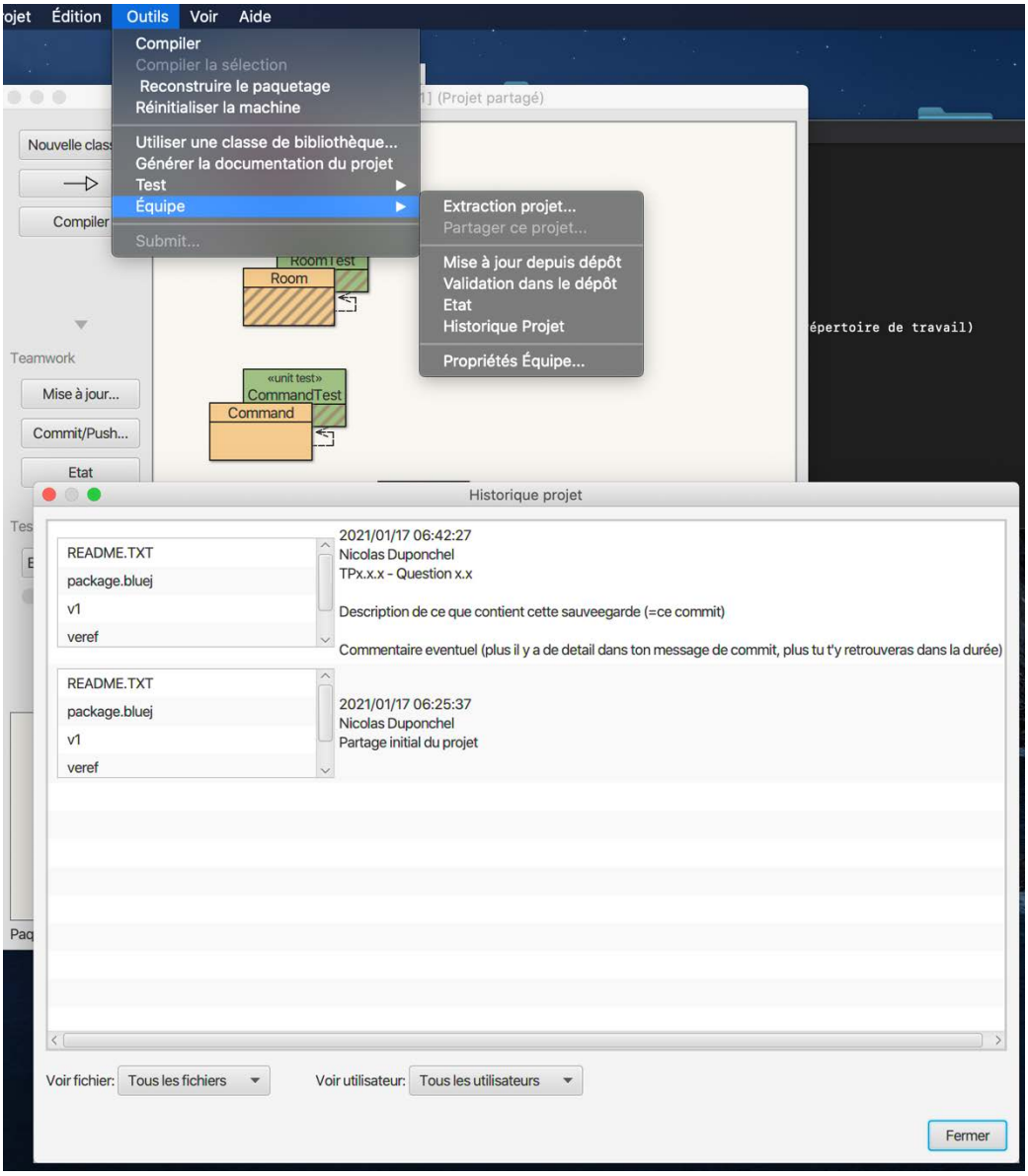


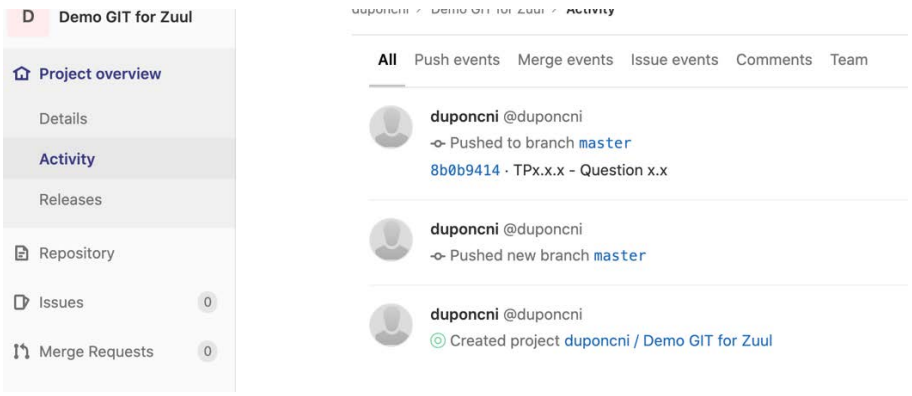
Une fois l'opération réussie, tu peux aller vérifier l'état de ton projet sur GitLab, consulter ton commit, et y voir les modifications apportées à chaque fichier :



Consulter l'historique git

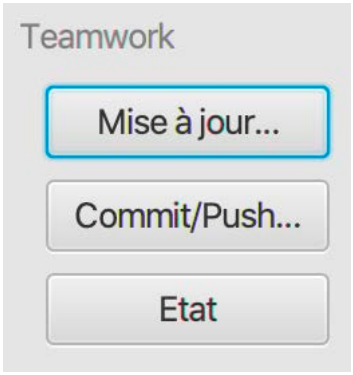
BlueJ offre la possibilité (tout comme tu peux le faire via GitLab) de visualiser l'historique de tes commits :





Mettre à jour son projet

Il est possible que des modifications soient apportées à ton projet sur ton GitLab et non sur ton projet local, par exemple lorsque tu es amené à travailler sur différentes machines, ou avec différentes personnes sur ton projet.
C'est pourquoi il faut **toujours s'assurer que ton projet LOCAL est à jour avant de commencer** un nouvel exercice. Pour cela, il faut que tu récupères sur ton projet local toutes les modifications qui ont pu être poussées sur GitLab. On appelle ça « puller » dans le jargon.



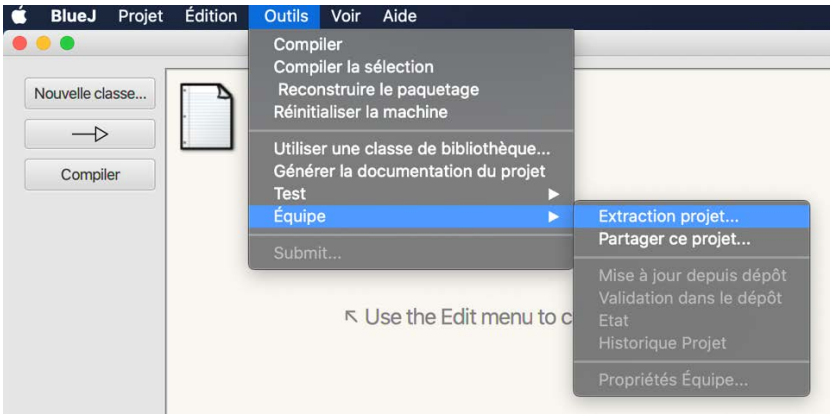
La plupart du temps, tu n'auras rien à mettre à jour.

A la fin de chaque session de travail, **NE PAS OUBLIER *Commit* + *Push***.

Si plusieurs exercices ont été faits pendant une même session, il est conseillé de faire un *Commit* à la fin de chaque exercice.

Récupérer son projet depuis GitLab

Voici un exemple de comment récupérer ton projet hébergé sur GitLab sur une nouvelle machine (éventuellement en cas de perte de projet en local par exemple).
Outils → Equipe → Extraction projet …



Remplis tes infos et valide. BlueJ va créer un nouveau projet local basé sur celui hébergé sur ton GitLab.

Supprimer un projet GitLab

Sur le projet : Settings → General → Advanced → Expand → (scroll end) Remove Project

Attention ! C'est irréversible. Il vaut parfois mieux en créer un nouveau (refaire le tuto et avoir une nouvelle URI de projet), et être sûr que tout se soit bien passé avant de supprimer définitivement le projet !

Liens utiles

- Tuto git by BlueJ : https://www.bluej.org/tutorial/git/
- Create SSH key : https://git.esiee.fr/help/ssh/README#generating-a-new-ssh-key-pair
- Pro Git : https://github.com/progit/progit2/releases/download/2.1.194/progit.pdf

Auteur 20/01/21: Nicolas Duponchel
(révision 09/02/21 : Denis BUREAU)