
Utilisation de git pour le travail en bin me

Le projet devra  tre r alis  par  quipe de 2 (bin me). Pour que le travail en  quipe se passe bien, il est bon d’avoir des outils pratiques et des m thodes de travail adapt es. Pour le d veloppement de logiciel on utilise en g n ral un syst me de gestion de version comme **Git**.

Voir aussi <https://nicolas.thiery.name/Enseignement/Info111/collaboration.html>

Voir aussi les aides sur la page web du cours.

Git est un logiciel de gestion de versions d centralis . Cr   par Linus Torvalds (l’auteur du noyau Linux), c’est actuellement le logiciel de gestion de versions le plus populaire. Vous l’avez utilis  sans le savoir   travers les outils **Travo** et **course.py**. Ce qui suit n’est *pas* un manuel d’utilisation de **Git**, mais une organisation simple du travail qui devrait vous permettre de travailler en bin me, sans risquer d’avoir des probl mes de conflit de modification. Ce n’est pas la bonne mani re d’utiliser **Git**, s’il l’on sait s’en servir. En particulier, nous n’utiliserons pas la fonctionnalit  principale de fusion de modifications.

Le d p t GitLab.

Quand vous d posez votre travail avec **course.py submit**, les donn es sont stock es dans un d p t **GitLab** h berg  par l’universit . Ce d p t est accessible dans votre navigateur sur :

<https://gitlab.dsi.universite-paris-saclay.fr/prenom.nom/L1InfoProgMod-AnneeScolaire-NomDuR pertoire>

Mon d p t pour le projet est, par exemple :

<https://gitlab.dsi.universite-paris-saclay.fr/florent.hivert/L1InfoProgMod-2022-2023-Projet>

Vous pouvez   tout moment consulter l’ tat de ce qui est d pos  sur ce site dans vos d p ts. Vous pouvez aussi y consulter et t l charger les fichiers   la main. Nous allons voir comment **Git** peut vous simplifier le travail en bin me.

Configuration de l’acc s.

Par d faut, quand on vous cr e un d p t (lors de votre premier **course.py submit**), il n’est accessible qu’  vous et   l’ quipe enseignante. Pour donner les droits   votre bin me, il faut suivre la d marche suivante :

1. S’assurer que le projet existe bien sur **gitlab** avec

```
cd ~/ProgMod  
./course.py submit Projet MonGroupe
```

Le binôme est composé de deux étudiants nommé ci-dessous A et B.

2. L'étudiant A donne l'accès à B

```
cd ~/ProgMod
./course.py share_with Projet prenomB.nomB
```

3. De son coté, l'étudiant B donne l'accès à A

```
cd ~/ProgMod
./course.py share_with Projet prenomA.nomA
```

4. Indiquez à GitLab que vous travaillez en binôme. Pour cela, l'un des deux membres, ici A, fait :

```
cd ~/ProgMod
./course.py submit Projet --leader_name prenomB.nomB
```

Nous ne corrigerons que le devoir de B : à charge pour vous de vous assurer que celui-ci contient bien l'intégralité du travail du binôme !

Votre binôme a maintenant accès en lecture à votre projet. Il peut consulter votre dépôt sur GitLab.

Récupération du projet du binôme.

Vous pouvez récupérer les fichiers de votre binôme sur le site à la main, mais on peut faire mieux avec `Git`.

Récupération à la main.

1. Se connecter sur le dépôt de votre binôme via un navigateur web grâce à l'adresse web indiquée plus haut ;
2. Cliquez sur le nom du fichier que vous voulez récupérer, puis cliquer sur la flèche en haut à droite (Download).

Pour télécharger tous les fichiers en format compressé, ne pas cliquer sur le nom d'un fichier en particulier, mais cliquer sur la flèche (Download) en haut à droite de la liste des fichiers et choisir le format compressé qui vous convient.

Récupération avec `Git`.

Ça se fait en deux temps :

1. **Faire un clone du dépôt de votre binôme (à ne faire que la première fois) :**

```
git clone https://gitlab.dsi.universite-paris-saclay.fr/prenom.nom/L1InfoProgMod-2022-2023-P
```

vous avez maintenant un répertoire portant le nom `L1InfoProgMod-2022-2023-Projet` qui contient le code de votre binôme. Vous pouvez renommer ce répertoire comme vous le souhaitez avec

```
mv L1InfoProgMod-Projet ProjetBinôme
```

2. **Récupérer les modifications de votre binôme (à faire chaque fois qu'il aura fait un submit)**. Allez dans le répertoire du projet de votre binôme :

```
cd ProjetBinôme
```

puis faire

```
git pull
```

Vous devriez avoir un message ressemblant à

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From https://gitlab.dsi.universite-paris-saclay.fr/florent.hivert/L1InfoProgMod-Projet
 4f4af71..64c35d5  master    -> origin/master
Updating 4f4af71..64c35d5
Fast-forward
 bla.cpp | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

qui m'informe sur les deux dernières lignes que dans le fichier `bla.cpp` une ligne a été supprimée et une ligne a été ajoutée.

Attention ! Vous pouvez copier les fichiers de votre binôme, mais **ne les modifiez pas !!!**. Vous risquez d'avoir des conflits. Si vous l'avez fait par erreur, vous pouvez supprimer le répertoire et refaire le clone selon l'étape 1 ci-dessus.

Quelques conseils pour travailler à deux.

- Répartissez vous le travail en évitant de travailler dans le même fichier, par exemple : l'un écrit les tests pendant que l'autre écrit le code.
- Dès que vous avez écrit une fonctionnalité qui marche, faites un submit pour la sauvegarder et que votre binôme puisse la récupérer.
- Sur **GitLab**, on peut voir l'historique des modifications en cliquant sur «branch». Le script `course.py` garde une information de la forme

Soumission depuis `a-145595.ups.u-psud.fr` par Florent Hivert

qui n'est pas informative. Si vous préférez mettre un message plus informatif, il faut aller dans le répertoire et faire :

```
git commit -am "Message décrivant la modification"
```

- Vous pouvez voir la liste des fichiers modifiés avec

```
git status
```

Les fichiers marqués «Untracked files :» sont ignorés par **Git**. Pour qu'il en tienne compte il faut faire

```
git add NomDuFichier
```

- Vous pouvez voir la liste des modifications

```
git diff
```

Les lignes supprimées sont en rouge, celles ajoutées en vert. S'il y a plusieurs pages de modifications, vous pouvez vous déplacer avec les flèches. Tapez «q» pour sortir quand vous avez fini.

- Bien sûr, si vous êtes curieux, vous pouvez explorer les fonctionnalités de **Git** qui est très puissant.