**Mémo GitHub / Git**

Créer un compte sur [https://github.com](https://github.com/)

Installer git : <https://git-scm.com/downloads>

Dans votre ligne de commande, tapez :

git config --global user.name "[Prenom] [Nom]"

git config --global user.email "[username]@[…]"

Afin de vous créer un clone de travail local, tapez :

git clone <https://github.com/Skabran/Robots-craseurs.git>

Cela a pour effet de créer un dossier [projectname] dans lequel vous pourrez travailler.

*Commandes git courantes*

git status

Permet de connaître l’état actuel de votre dépôt local : liste des fichiers sous contrôle contenant des modifications, liste des fichiers présents et non contrôlés, etc.

git remote -v

Permet de voir la liste des dépôts distants connus (“origin” est le nom du dépôt distant depuis lequel le dépôt local a été cloné)

git diff [file]

Permet de voir les modifications apportées à un fichier et non encore commitées

git checkout -- [file]

Permet d’annuler les modifications apportées à un fichier et de le remettre dans l’état où il était précédemment

git add [file]

Permet aussi bien :

* d’ajouter un fichier non contrôlé au dépôt lors du prochain commit
* d’ajouter les modifications d’un fichier sous contrôle au prochain commit

git reset HEAD [file]

Permet d’enlever un fichier de la liste des fichiers à ajouter au prochain commit

git commit -m "…"

Permet d’ajouter un commit au dépôt (sous réserve d’avoir fait au moins un add avant). Cette commande permet de saisir le message du commit et voir au passage la liste des fichiers concernés

git pull

Récupère les commits du dépôt d’origine dont le dépôt local ne dispose pas encore

git push

Envoie vers le dépôt d’origine les commits dont il ne dispose pas encore (vous pouvez constater sur l’interface web GitLab que vos nouveaux commits ont bien été ajoutés)

*Mise en commun du travail*

Une fois que vous avez créé des commits dans votre dépôt de travail (git commit) et envoyé ces commits dans votre dépôt distant (git push), vous allez pouvoir les proposer pour intégration au sein du dépôt commun du groupe.

Pour cela, sur GitLab, vous allez créer une “Merge Request” depuis votre dépôt à destination du dépôt du groupe.

Allez sur la page de votre projet, puis dans l’onglet “Merge Requests”.

Cliquez sur “New Merge Request” :

* à gauche, la source est normalement déjà correctement configurée sur votre dépôt [username]/[projectname]. Il vous suffit de sélectionner la branche à utiliser. Votre dépôt ne dispose que de la branche “master” que vous allez donc sélectionner.
* à droite, la cible est normalement déjà configurée sur la branche “master” du dépôt du groupe ([groupname]/[projectname]).

Cliquez sur “Compare branches and continue”.

Vous pouvez alors donner un titre, et une description à la Merge Request. Vous pouvez également consulter l’ensemble des modifications soumises.

Si vous êtes satisfait, cliquez sur “Submit merge request”, sinon vous pouvez continuer de travailler, rajouter de nouveaux commits et soumettre à nouveau une demande d’intégration ultérieurement.

Votre nouvelle requête est maintenant visible dans l’onglet “Merge Requests” de la page du dépôt du groupe. Le contenu de la requête est visible par l’ensemble des membres du groupe qui peuvent alors entamer une discussion au sujet de la requête. Si nécessaire, tant que la requête est ouverte, l’auteur de la requête a la possibilité d’ajouter de nouveaux commits à son dépôt, qui seront ajoutés à la requête en cours.

Si la requête n’est pas en conflit avec l’état actuel du dépôt du groupe, le chef de projet a la possibilité d’accepter la requête. Pour que seul le chef de projet puisse accepter les Merge Requests, il faut *protéger* la branche master du dépôt du groupe. Pour cela, le chef de projet peut aller dans le menu “roue crantée” dans la section “Protected branches”, sélectionner la branche master puis choisir “Allowed to merge : Masters” / “Allowed to push : No one”. De cette manière, seul le chef de projet pourra accepter les Merge Requests et la seule manière d’ajouter des commits à ce dépôt sera par l’intermédiaire d’une Merge Request.

Une fois une requête acceptée, tous les membres du groupe peuvent récupérer les nouveaux commits et les intégrer à leur dépôt de travail (git pull).

Si plusieurs Merge Requests sont à l’état de demande et sont toutes acceptables, il se peut que l’acceptation de l’une en mette une autre en conflit. L’auteur de la Merge Request en conflit peut alors simplement :

* récupérer les commits qui lui manquent (git pull)
* résoudre les conflits (modifier les fichiers concernés et les ajouter pour le prochain commit)
* faire un commit

Ce commit sera alors ajouté à sa Merge Request en cours qui, si tout va bien, redeviendra alors acceptable par le chef de projet.