# **GITHub**

**Configuration :**

git config --global user.name "Votre nom ou pseudo"

git config --global user.email [Votre@email.com](mailto:Votre@email.com)

**Pour récupérer un dossier sur git  (mode http url) :**

**git clone** [**https://github.com/inkaran/test.git**](https://github.com/inkaran/test.git)

**ou crée un repository sur git, et envoyer son projet sur git :**

git init

git remote add origin <https://github.com/inkaran/nomdurepertoire.git> (faire un add et commit d’initilisation avant)

git push -u origin master

git add votreficher ou –A ( pour tous les ajouter)

**Pour désindexer :** git reset HEAD votrefichier

**Après avoir fait des modifications, faire un commit avec un message**

git commit -m « Message du commit »

**Pour enregistrer tous les fichiers de l’index et commit en une commande :**

git commit –a –m « message »

**Pour l’envoyer sur le dépôt (=remote) :**

git push origin master (ou branche )

**Pour récupérer les derniers modification du projet/répertoire de Github et fusionne avec le projet/répertoire local :**

git pull

**Pour voir l’état (changement fait, fichier indexé ou pas) :**

git status

**Pour voir aspect visuel de l’arbre et toutes les informations :**

gitk

**Pour afficher tous les commits realisés et q pour la quitter :**

git log

**Pour se placer au niveau d’un commit il faut rentrer son « SHA » :**

git checkout SHADuCommit

**Revenir à la branche principale :**

git checkout master

**Modifier le message dernier commit :**

git commit --amend -m "Votre nouveau message"

**Savoir sur qu’elle branche on se trouve (HEAD):**

git branch

**Créer une nouvelle branche :**

git branch « nouvelle branche »

**Se placer dessus :**

git checkout nomDeLaBranche

**Crée une nouvelle branche et se placer dessus :**

git checkout –b « nouvelle branche »

**Supprimer une branche (après avoir merger):**

git branch –d nombranche

**Se placer sur master et copier le contenu de la branche1 :**

git checkout master

git merge branche1

Voir les branches qui ont fusionné :

git branch --merged

**Gérer les conflits :**

git mergetool --tool-help ( pour voir les outils supporter pour gérer le merge)

git mergetool –tool=emerge ( pour assigner un outils disponible par défaut )

git mergetool nomficher ( pour voir comparer les 2 fichiers en question )

Dans emerge : **d a** pour garder fichier a et **d b** pour garder fichier b.

**Pour ignorer des fichiers dans git pour raison de sécurité :**

Tous les fichiers de configuration (config.xml, databases.yml, .env...)

Les fichiers et dossiers temporaires (tmp, temp/...)

Les fichiers inutiles comme ceux créés par votre IDE ou votre OS (.DS\_Store, .project)

**On les place dans .gitignore, on liste les noms des fichiers et on l’ajoute à l’index**

**Pour mettre les modifications de côté (sans faire de commit) et faire autre chose à coté :**

git stash / git stash pop (pour le récupérer)

# Dans GitHub

**Fork :** permet de récupérer une copie du projet dans son github (puis clone pour le récupérer dans son bureau)

**Full request :** permet après avoir récupérer le projet travailler dessus de proposer les modifications à l’auteur et si il accepte il merge.

**Issue :** liste de chose à faire, de bugs, d’améliorations qu’on peut commenter. On peut assigner un travail à qq’un pour éviter que 2 personnes travaillent sur la même chose.

**Gists :** partager des morceaux de code en ligne.

# Commande annexe :

**Supprimer un fichier :**

git rm nomfichier

**Renommer un fichier (le modifie, supprimer l’ancien et ajoute le nouveau à l’index) :**

git mv nom\_origine nom\_cible

**Pour renommer un remote :**

git remote rename nomOrigine nomDésiré

**Pour ajouter un dépôt d’une autre personne :**

git remote add nomDésiré urlDuDepot