**Git Procédure de Migration de sources dans SGitHub**

**- Avec Jenkins F git ile -**

ESTIMATION DU TEMPS : 30 MIN

1. (DEV) Communiquer à tous les contributeurs du repo la date de la migration en amont pour éviter d’apporter des modifications au repo le jour J.
2. (DEV) Vérifier que les repos sur **GitBucket** et **SGitHub** sont bien synchronisés.

Pour chaque branche, vérifier que les nombres de commit sont identique

|  |  |
| --- | --- |
| SGitHub |  |
| GitBucket |  |

1. (DevOps) Vérifier que le mode **Miroir** est désactivé / Désactiver le mode miroir le cas échéant :

**REMARQUE :** Si la SQUAD n’a pas les droits d‘écriture sur le repo en question au niveau de SGitHub, donc :

 => Mettre à jour le miroir depuis GitBucket avant de continuer

|  |
| --- |
|  |

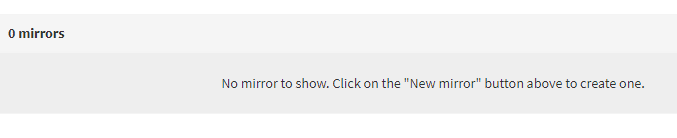
Sinon, procéder à la suite du MODOP 😊

**ATTENTION :** Si la SQUAD a déjà commencé à travailler sur SGitHub, il ne faut absolument pas faire des modifications du coté GitBucket, car ça risque d’écraser tout votre travail sur SGitHub.

* Connecter vous sur GitBucket en tant que « root »
* Sélectionner le repo à déplacer
* Cliquer sur « Mirrors » :

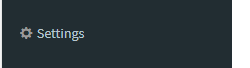


* Vérifier qu’il n’y a plus de Miroir de ce source :

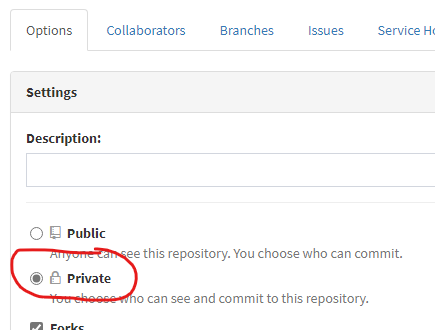


1. (DevOps) Déplacer les sources GitBucket dans **MIGSG**.

* Connecter vous sur GitBucket en tant que « root »
* Sélectionner le repo à déplacer
* Cliquer sur « Settings » :



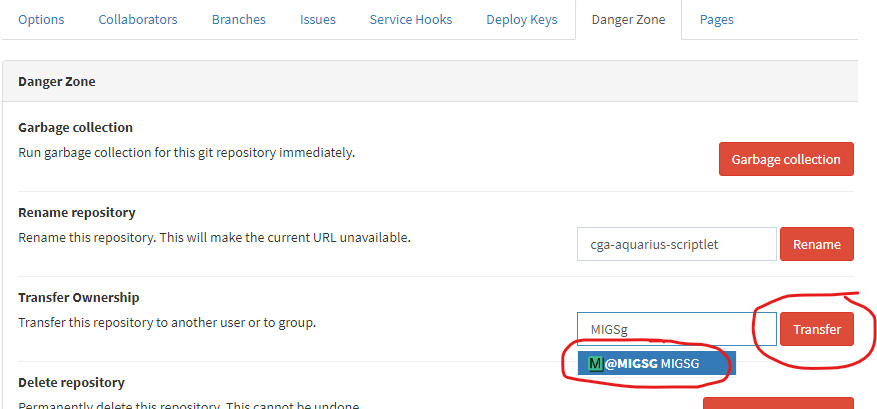
* Dans l’onglet « Options », changer le type de repo en « private » :



* Puis appliquer les changements en cliquant sur « Apply changes » :



* Puis dans l’onglet « Danger Zone », changer le propriétaire du repo vers « MIGSG » :



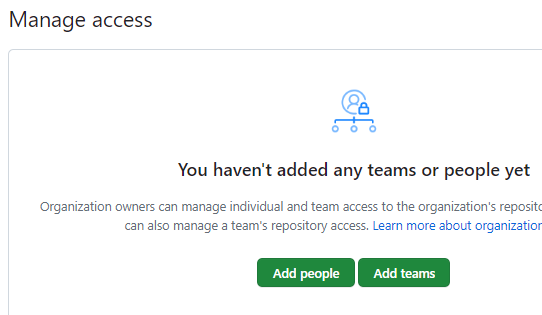
Confirmer le transfert par « OK »

1. (DevOps) Modifier les droits d’accès au repo sur SGitHub :

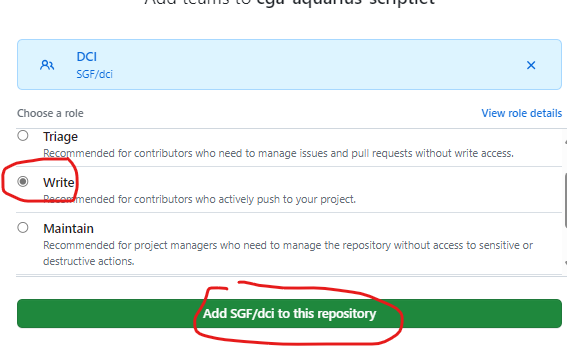
* Sélectionner les « Settings » du repo à modifier
* Sélectionner l’option « Collaborators ans teams » :



* Cliquer sur « Add teams » :



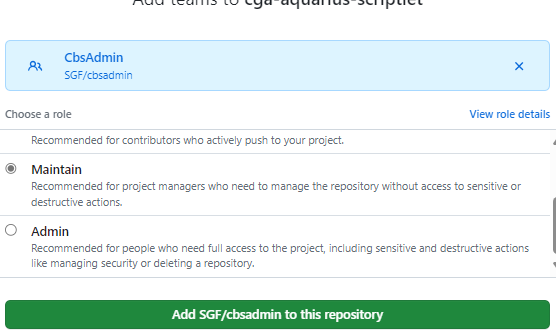
* Sélectionné l’équipe propriétaire du repo, et lui affecter les droits « Write » :



* Cliquer de nouveau sur « Add teams » :



* Sélectionner l’équipe Administrative du repo, et lui affecter les droits « Maintain » :



1. (DEV) Configurer son IDE, afin qu’il pointe vers SGitHub :

* Générer un Token personnel :
  + Aller sur « settings » :

|  |
| --- |
|  |

* + Cliquer sur « Developer settings »

|  |
| --- |
|  |

* + Générer un « Classic Token » => Cliquer sur « Generate new token » :

|  |
| --- |
|  |

* + Utiliser votre G comme nom du Token, sélectionner les scopes d’utilisation du Token puis cliquer sur « Generate ».

|  |
| --- |
|  |

* + Sur votre IDE, configurer le proxy :

=> git config --global http.proxy http://gXXXX:YOURPASSWORD@proxy.cgasg.local:3128

* + Ajouter le repo distant :

=> g remote -v

=> g remote add github  “l’url sgithub du repo”

* + Changer le origin url :

=> git remote set-url origin “l’url sgithub du repo”

* + Faire un pull. Il vous sera demandé de saisir son ID et mot de passe SGitHub => Utiliser le token que vous avez générer :

|  |
| --- |
|  |

1. (DEV) Sur SGitHub, modifier la configuration des fichiers pom.xml et les fichiers Jenkinsfile

|  |
| --- |
|  |

* Mettre à jour toutes les url du dépôt Gitbucket vers le nouveau dépôt SGitHub

POM.XML :

|  |
| --- |
|  |

|  |
| --- |
|  |

Jenkinsfile :

|  |
| --- |
|  |

1. Mettre à jour la configuration du Job sur Jenkins :

* Connecter vous sur Jenkins
* Sélectionner le Job lié au Pipeline en question.
* Cliquer sur « Configurer ».

|  |
| --- |
|  |

* + Dans la partie « Pipeline », mettre à jour le « Repository URL » par le nouveau URL SGitHub.
  + Dans « Credentials », choisir le token « MyGitHubRepoToken/\*\*\*\*\*\*(SGitHub Integration SGF) ».

|  |
| --- |
|  |

* Cliquer sur « Appliquer » puis « Sauvegarder »,

|  |
| --- |
|  |

* Lancer le build.

|  |
| --- |
|  |

*That’s all Folks!*