

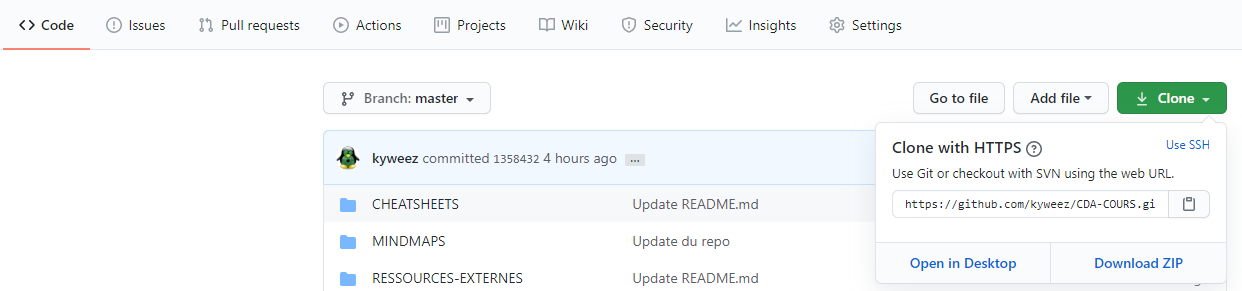
|  |
| --- |
| Gestion de branches avec flow |
|  |
| 26 juin  Concepteur Développeur D'Application  Créé par : Jonathan Jeanniard & Aurélien BOUDIER |

1. A quoi sert git flow ?

Git Flow est une méthode, une architecture Git permettant de séparer au maximum le travail et de toucher le moins possible à la branche Master. Cette méthode représente donc une architecture en branches. Pour commencer, on aura la branche Master au moment de l’initial commit. Ensuite, on va créer une branche develop, qui sera la branche principale. C’est par ici que vont passer tous les commits. La branche master ne récupèrera le code de la branche develop qu’au moment d’une release, et à ce moment elle sera tag afin de différencier une version.

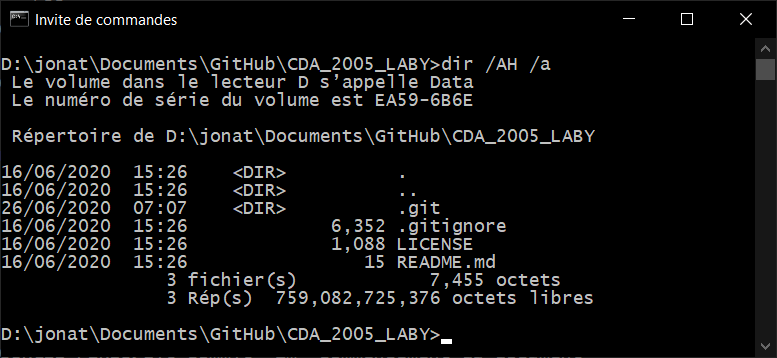
1. Récupération du dépôt

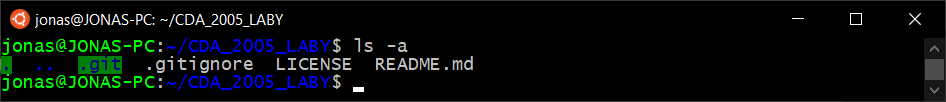
Dans un premier temps il faut récupérer un dépôt quel qu'il soit avec git clone.



1. Vérification git

Ouvrez l’invite de commande Windows (cmd) ou un terminal Linux. Naviguez jusqu'au répertoire où se trouvent le dépôt cloné avec la commande "cd" pour les deux systèmes.





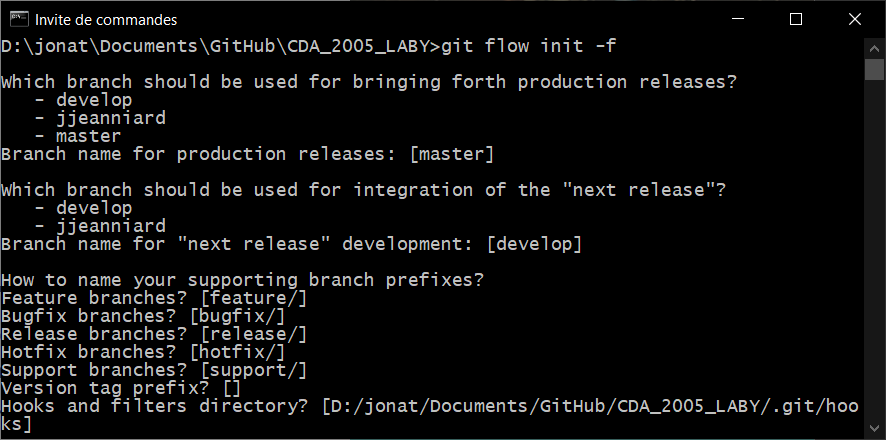
Une fois dans le répertoire du dépôt vérifiez que le dossier caché de git est bien présent :

|  |
| --- |
| Windows : dir /AH/a |
| Linux/Mac : ls -a |

1. Initialisation de flow

A partir de là, les commandes de git sont communes au 2 systèmes, nous utiliserons le cmd de Windows pour la suite pour plus de facilité.

Donc pour initialiser la gestion des branches, on tape git flow init -f .

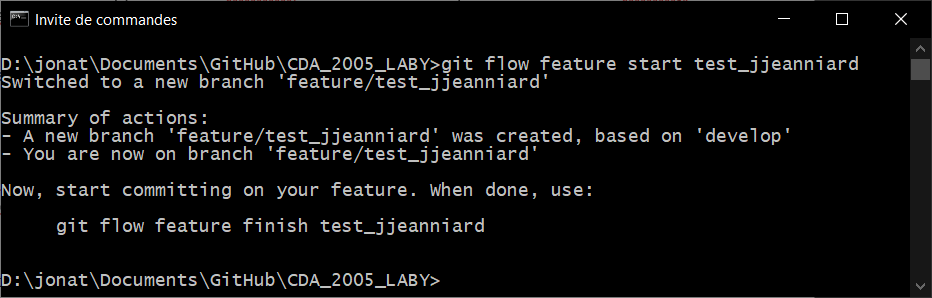


De nombreuse questions vous sont posées, vous pouvez tout mettre par default en appuyant sur Entrée (si un problème survient au moment de "next release", il faut au préalable créer la branche "develop"). Chaque branche correspond à une fonction donnée, comme on peut le remarquer.

1. Première branche

La première fonctionnalité de notre programme sera "test", donc la commande pour réaliser notre espace de travail est "git flow feature start test\_jjeanniard".

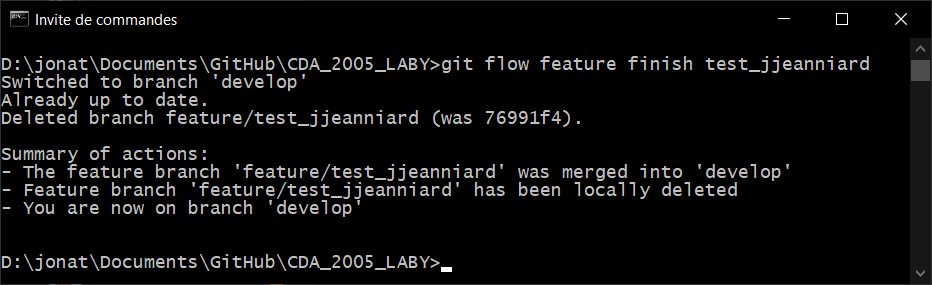
Par convention nous mettrons le pseudo/nom, précédé d'un underscore après le nom de la fonctionnalité. Cette commande va créer une nouvelle branche tout en récupérant ce qui se trouve dans la branche "develop" et puis changera automatiquement de branche en nous plaçant dans la nouvelle.



NB : Dans les livres, ils préconisent de commencer dans la branche "feature" avec pour nom "init\_projet" pour la réalisation d'un nouveau projet.

1. Fin de la branche

Après de nombreux commits, pushs, du "code review" sur la fonctionnalité (cycle sans fin) et un "pull request" réussi sur Github. Il est temps de changer de branche, mais avant d'en créer une nouvelle, il est préférable de « rendre la liberté » à votre précédente branche avec la commande "git flow feature finish test\_jjeanniad".



Un merge de votre branche se fait automatiquement, avant d'être supprimé et puis vous êtes de nouveau switch dans la branche "develop". Une nouvelle vie de branche peu commencer (feature, hotfix, bugfix…).

Une chose à savoir, il n'est pas interdit de travailler sur plusieurs branches. A vous de savoir celles-ci et de ne pas modifier le mauvais fichier avec la mauvaise branche.

Crédit :

|  |  |  |  |
| --- | --- | --- | --- |
| Action | Auteur | Descriptif | Date |
| Réalisation du document | Jonathan JEANNIARD | Découverte de la commande git flow | 26/06/20 |
| Agrémentation du document | Aurélien BOUDIER | Correction, reprise de la forme et ajour d’une section | 26/06/20 |