Tutoriel Github

Présentation globale :

Github est un service web permettant de gérer et de développer tout type de projet lié à du code informatique via le logiciel de gestion de versions Git, qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées sur ces derniers.

Il assure un accès libre ou privé, et permet une collaboration commune et directe sur les différents projets, et ce pour un nombre illimité d'utilisateurs.

En janvier 2020, Github comptabilisait plus de 40 millions d'utilisateurs avec plus de 190 millions de projets stockés sur le site. Ce qui en fait le site numéro 1 dans son domaine. Il est devenu un incontournable pour les projets informatiques en groupe.

Les fonctionnalités Git permettant les gestion des différents dépôts sont très nombreuses et permettent toute sorte de modification, nous allons ici nous intéresser aux basiques qui assurent une utilisation simple :

Les branches :

Les branches sont ce à quoi le système Git doit sa popularité et son efficacité, elles permettent de pouvoir travailler sur différentes versions du projet, voici par exemple à quoi ressemblerait le projet « A » et les variantes (donc branches) pouvant en découler :

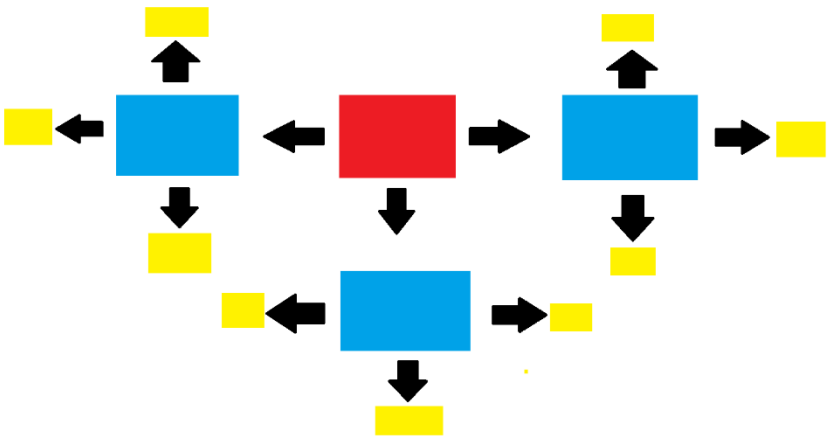


Schéma personnel

Avec en rouge le projet A, en bleu le projet A.1, A.2 et A.3, en jaune A.1.1, A.1.2 jusque A.3.3 etc…

Conflits :

Lorsque deux utilisateurs travaillent sur le même projet et sur la même branche en simultané, le fait de modifier le code sur la même ligne peut engendrer des erreurs étant donné que le service web ne saura pas laquelle des 2 versions modifiées doit être considérée comme la bonne.

Ce genre d'erreur peut aussi survenir lorsqu'un un fichier sur lequel un utilisateur travaille se retrouve supprimé.

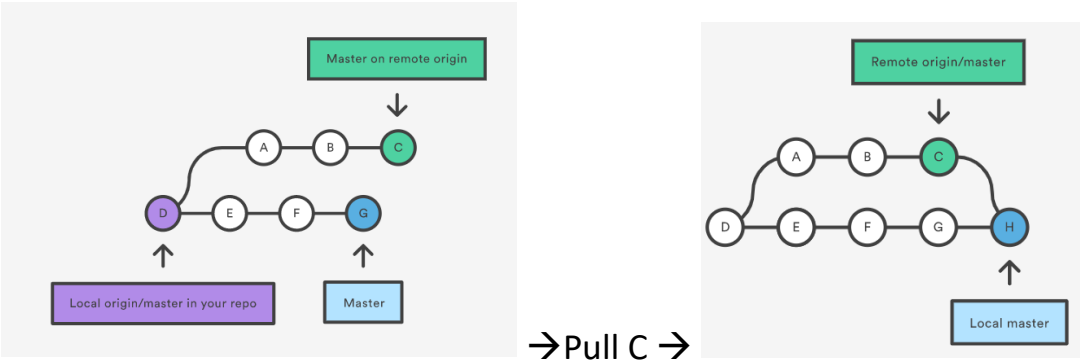
Le fichier sera alors considéré comme « en conflit » et ce sera aux utilisateurs de résoudre le problème.

Commit :

Git-commit est la commande permettant d'enregistrer les modifications venant d'être réalisées dans le dépôt. Il est conseillé de souvent l'utiliser étant donné que comme vu précédemment, à chaque « sauvegarde », la version antérieure reste disponible, ce qui peut s'avérer très utile en cas d'erreur.

Pull :

Git pull permet le téléchargement d'un dépôt distant (donc non présent directement sur la machine de l'utilisateur) comme par exemple récupérer les différentes versions d'un même code pour pouvoir travailler dessus aussi, comme illustré dans les schémas ci-dessous :



Crédits : atlassian.com

Push :

Git push assure l'upload d'un projet ou d'un code présent provenant de la machine locale de l'utilisateur sur le service web, entre autres cette fonctionnalité permet le transfert d'un dépôt local à un dépôt distant.

Dépôt local et remote :

Un dépôt local et un dépôt présent directement sur la machine de l'utilisateur, il peut donc être modifié et ce, sans accès internet.

La commande remote permet de créer le dépôt distant (donc présent sur le serveur web) sur lequel on souhaite déposer les fichiers provenant de notre machine.

Pull request :

Une Pull request permet de proposer un changement dans un projet présent sur Github, proposition soumise à vérification par les chefs du projet en question, ce qui assure une collaboration sûre entre les différents utilisateurs (pas de sabotage intempestif etc…)

Merge :

Git merge permet d'intégrer des branches à une branche ciblée, on pourrait réutiliser le schéma pour Git Pull (en enlevant les légendes, H serait une partie dans laquelle C et G se rejoignent).

Autres commandes en ligne :

Il reste bien sûr de nombreuses fonctionnalités comme par exemple git switch qui permet d'alterner de branche comme on le souhaite ou encore git rm pour la suppression de fichier.

Toutes ces fonctions permettent donc une navigation et utilisation pratique à travers toute la mise en œuvre d'un projet informatique qu'il se fasse en solitaire ou en groupe, à l'école ou en entreprise.