Master 1 : Big Data

Bosli Maria

Outils libres pour le développement logiciel

1. Définir le workflow gitflow : C'est un modèle de branche , pour ajouter des commandes qui facilite le travail, et pour ne pas utiliser des mauvaise branche ou d'oublié de merger . On l'utilise pour séparer au maximum le travail et de toucher le moins possible à la branche Master(comme on a déjà dit c'est une architecture de branche).
2. Les avantages du workflow gitflow : Permet de mettre en place tout de suit des bonnes pratiques de dev et préparer la mise en place des logiciel qui permet d'automatiser progressivement les développements , et aussi de déminuer le risque de désorganiser tout le système.
3. Les inconvénients du workflow gitflow : il est difficile à gérer dans un modèle de déploiement continu (il fonctionne bien pour les produits dans un modèle ou les versions sont effectuées une fois toutes les quelques semaines, mais que ce processus de décompose considérablement lorsque on publie une fois par jour ou plus).
4. La définition et l'utilité des branches :
    - Feature : elle est utilisée pour développer une nouvelle fonctionnalité. Cette branche est toujours tirée depuis la branche develop. Une fois terminée, on fait le merge sur develop.
    - Hotfix : elle est utilisée pour réparer un bug urgent en production. Cette branche est toujours tirée depuis la brqnche master. Une fois on termine on fait le merge sur master et aussi sur develop.
    - Realise : elle est utilisée pour produire une nouvelle version de son projet. Cette branche est toujours tirée depuis la branche develop . Une fois terminer on fait le merge sur master et seu develop car il est possible de faire des commits sur cette branche pour résoudre des bugs par exemple.
    - Master : la branche par défaut sur git s'appelle master .
    - Develop : elle est crée à partir de main. Une branche release est crée à partir de develop . des branche feature sont aussi crée à partir de cette dernière, et lorsqu'une fonctionnalité termine elle est mergée dans la branche develop.
5. Les commandes git pour crée un tag sachant qu'on est dans develop :
   On doit se mettre dans la branche develop et en suite on fait :
   Git tag -a <tag_name> -m "msg"
   Git push -tags
6. -On commence par créer la branche du hotfix :
   git checkout -b hotfix-x.x.x master
   -Ensuite,on fait notre correction git commit -m "fixer les prblm"
   -Enfin on commit dans master et dans develop :
   git checkout master
   git merge --no-ff hotfix-x.x.x
   git tag -a x.x.x -am "message"
   git checkout develop
   git merge --no-ff hotfix-x.x.x
   -et la on supprime la banche du hotfix
   git branch -d hotfix-x.x.x

7. la commande git à exécuter après la validation de la branche release pour passer en prod :

git merge release prod

8. la commande git stash sert a enregistrer l'état actuel du répertoire de travail e de l'index , et on revient a un répertoire de travail propre. Du coup elle enregistre nos modification en locale et rétablit le répertoire de travail pour qu'il correspond au commit head. La commande pour retourné en arrière de git stash c'est git stash pop