

1- Définir le workflow gitflow, pourquoi on l'utilise

Le workflow gitflow est un modèle qui a pour but de modifier le moins possible la branche principale en travaillant, en fonction de ce qu'il y a faire, sur les branches concernées. En général, il y en a 5 : feature, develop, release, hotfixes et master. On utilise alors plusieurs branchements pour faire évoluer le code. Il est utilisé principalement pour avoir une bonne séparation des versions et des différentes évolutions tout en évitant les conflits souvent créés dans les travaux d'équipes.

2- Quels sont les avantages du workflow gitflow

La branche master est toujours récupérable et représente une version à part entière du livrable. Le travail en équipe est censé être simplifié et les conflits sont en majorité évités.

3- Quels sont les inconvénients du workflow gitflow

Ce type de modèle peut poser problèmes pour les livrables en continu, par exemple avec les productions agiles, on peut se retrouver avec une très longue branche develop et plusieurs passages dans les branches feature et release, mais avec très peu de version master, ce qui veut dire que si des hotfixes sont mis en place, les nouvelles fonctionnalités ne les prendront pas en compte avant longtemps.

4- Définir et donner l'utilité des branches : Feature, Hotfix, Release, Develop, Master

Feature : Branche destinée aux fonctionnalités à part entière. Elles sont travaillées à part puis ajoutées dans la branche develop.

Hotfix : Pour gérer les conflits détectés dans les versions sur la branche master.

Release : Là où on récupère les modifications que develop a eu avant de soit repasser dans la branche develop, soit déployer dans la branche master.

Develop : La branche où sont tous les développements. Après avoir ajouté les différentes fonctionnalités, avec la branche feature, ou renvoyer depuis la branche release.

Master : La branche principale. Contient toutes les versions finalisées de la production.

5- Donner les commandes git pour créer un tag, sachant que vous êtes sur la branche Develop

```
git switch master
```

```
git tag '2.0'
```

6- Vous êtes sur une branche Feature en train de finaliser un développement, on vous informe qu'il y a un bug de prod à corriger très rapidement. Donner les commandes git pour corriger le problème de prod en respectant le workflow git.

(Dans notre cas, on considère que les branches hotfixes et develop ont déjà été créé).

```
git switch master
git switch hotfixes
(on règle le bug).
git switch develop
git merge hotfixes
git switch master
git merge hotfixes
```

7- Donner les commandes git à exécuter après la validation de la branche release pour passer en prod

git switch master

git merge release

8- A quoi sert la commande git stash, donner la commande qui permet d'avoir un retour arrière de git stash

git stash permet de sauvegarder un état de travail, par exemple avant un pull. C'est utile pour remettre un travail au propre si besoin ultérieurement.

Pour faire un retour arrière : git stash drop