GitFlow

Les branches du GitFlow

La branche master : c'est la branche de production.
Chaque merge sur cette branche donnera lieu à une nouvelle version :

- Mineure pour les merges de hotfix
- Majeure pour les merges de release

Ce qui sera mergé sur cette branche devra faire preuve de la plus grande attention et sera de la responsabilité unique du lead développeur.

La branche dev : C'est la branche de développement qui recevra les fonctionnalités achevées, revues, documentées, testées et approuvées par le lead développeur.

Les branches feature : Ce sont des branches éphémères.
Chacune sera consacrée au développement d'une fonctionnalité précise. Une fois la fonctionnalité achevée, revue, documentée et testée, le développeur qui l'a traitée proposera un merge de sa branche sur la branche dev au lead dev.

- Si le merge est accepté et réalisé, la branche feature est alors supprimée.
- Si le merge est refusé, le lead dev indique les points à améliorer par le développeur avant qu'il puisse la proposer à nouveau au merge.

La branche release : C'est la branche qui permet de développer et tester une nouvelle version de l'application avant sa mise en production.

La branche hotfix : C'est une branche destinée à résoudre des bugs de la branche master.

Création d'une nouvelle fonctionnalité

Création d'une branche
Les développeurs travaillent sur les branches feature en partant de la branche dev.

- Créer une nouvelle branche en se plaçant dessus à partir de la branche « dev » :

  git checkout -b feat/myNewFeature dev
- Faire un premier commit vide pour activer cette branche :

  git commit -m « feat/myNewFeature : new branch » --allow-empty

Développement de la fonctionnalité

Commiter les changements importants (en suivant les règles de commit)

```
git add <file name>   #ajouter le fichier au stage
git commit            #commiter les changements
```

A moins d'un commit évident, on évitera la commande git commit -m « mon message ».
On préfèrera la commande git commit.
Une fois dans l'invite de commande, on écrira :

- Un titre explicite (49 caractères maximum) : feat/myNewFeature : titre du commit
- Un texte de description (pertinent et clair)

Revue de la fonctionnalité

Une fois la fonctionnalité terminée, le code doit être revu.

Merge de la fonctionnalité sur la branche dev

- Envoyer votre code sur le dépôt distant github
  git push origin feat/myNewFeature
- Proposer une pull-request au reviewer
  - Si la pull-request est refusée, effectuer les modifications attendues
  - Si elle est acceptée, elle sera mergée syr la branche develop.

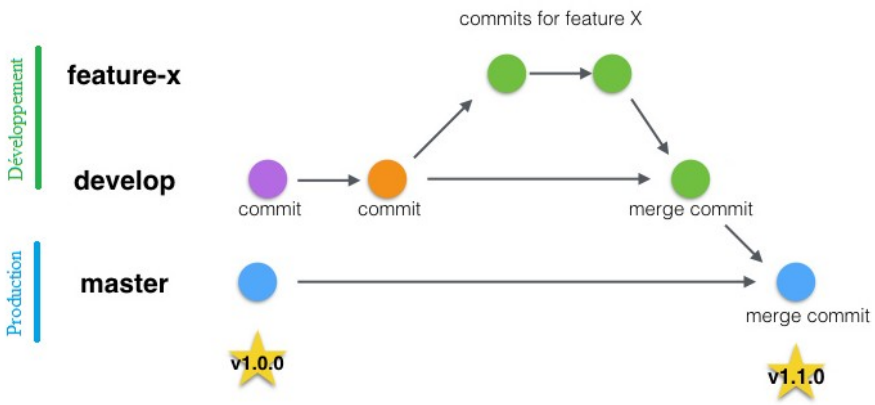
```
git checkout develop                    # se placer sur la branche develop
git rebase feat/myNewFeature            # linéariser l'historique
git merge feat/myNewFeature      ou     # merger sur develop
                                        → Résoudre les conflits éventuels
git branch -d feat/myNewFeature         # supprimer votre branche
```

Particularité des hotfixs et des releases :

Pour résoudre une hotfix ou créer une realease, l'esprit et la méthode sont identiques avec quelques variations listées dans le tableau ci-dessous.

	hotfix	**release**
Création de la branche à partir de	master	dev
Merge de la branche sur	dev et master	dev et master
Ajouter un tag de version	git tag V.1.2	git tag V.2.0

Un gitflow simple


commits for feature X
Développement
feature-x
develop
commit
commit
merge commit
Production
master
merge commit
v1.0.0
v1.1.0