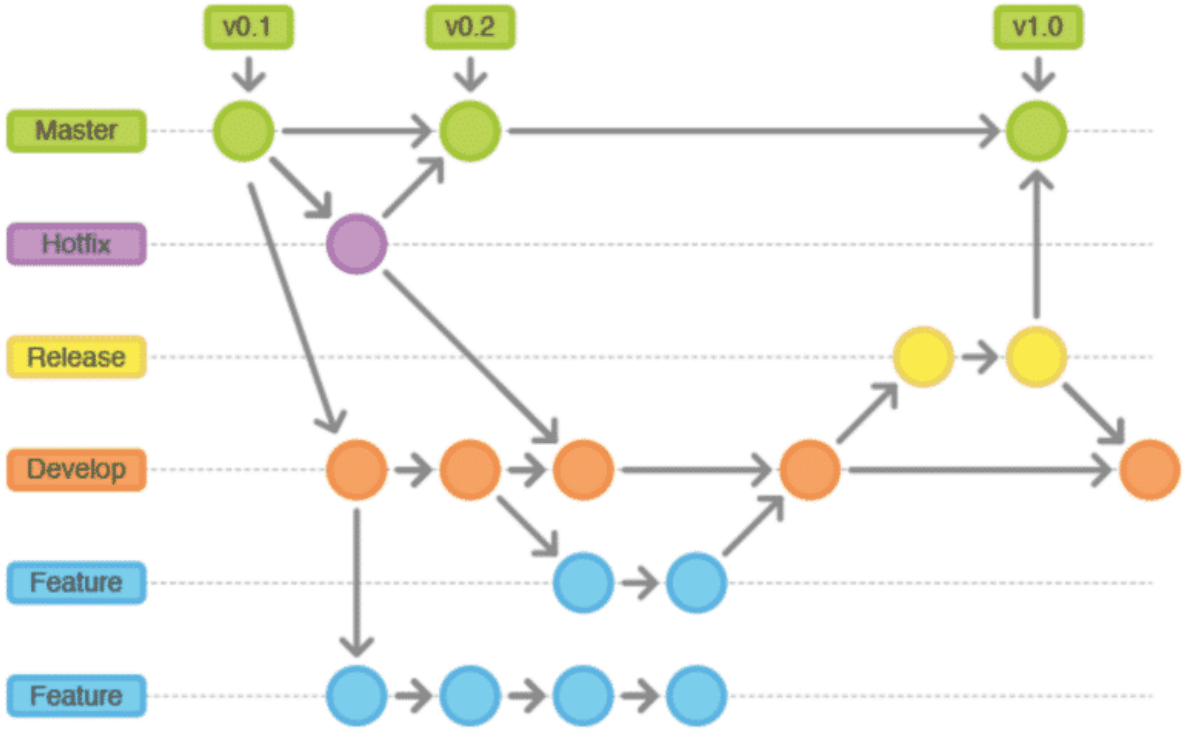
Gitflow

Pour travailler en équipe efficacement nous mettrons en place le git flow ci-dessous :



Utilisation de chaque branche

La branche master : c'est la branche de production sur laquelle on poussera nos modifications que pour une mise en production (ce qui y sera poussé devra faire preuve de la plus grande attention).
Chaque merge sur cette branche donnera lieu à une nouvelle version :

- mineure pour les merge de hotfix
- majeure pour les merge de release

La branche dev : C'est la branche de développement qui recevra les fonctionnalités achevées et testée.

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

Les branches feature : Ce sont des branches éphémères. Chacune sera consacrée au développement d'une fonctionnalité précise. Une fois la fonctionnalité achevée et testée, la branche sera poussée sur la branche dev et sera supprimée.

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

Méthode de travail

On ne touche jamais à la branche master. Seul le lead dev y est autorisé à merger une nouvelle branche.
Les développeurs travaillent sur les branches feature en partant de la branche dev.

Créer une nouvelle fonctionnalité

- Créer une nouvelle branche en se plaçant dessus : git checkout -b feat/myNewFeature
- Faire un premier commit pour activer cette branche :
  git commit -m « new branche feat/myNewFeature » --allow-empty

- Développer la fonctionnalité en commitant les changement important (voir règle de commit)
  git add <file name> #ajouter le fichier au stage
  git commit #commiter leschangement
- Une fois la fonctionnalité achevée, vérifier la qualité et la performance du code
- Envoyer votre code sur le dépôt distant github
  git push origine feat/myNewFeature
- Proposer une pull-request au lead dev
- Effectuer les modifications attendues si la pull-request est refusée
- Le lead dev mergera la pull-request si elle est acceptée
- Si le lead dev vous demande de merger la pull-request
  git checkout dev #se placer sur la branch dev
  git rebase feat/myNewFeature #linéariser l'historique
  git merge feat/myNewFeature (si nécessaire) #merger votre branche sur la branche dev
    - Résoudre les conflits éventuels
  git branch -d feat/myNewFeature #supprimer votre branche

Que ce soit pour créer une nouvelle fonctionnalité ou résoudre un bug l'esprit est le même et la méthode varie peu.

Règle de commit

A moins d'un commit évident, on évitera la commande git commit --amend pour commiter.
On préfèrera la commande git commit.
Une fois dans l'invite de commande, on écrira :

- Un titre pas trop long (49 charactères maximum)
- Un texte de description