GitFlow exemple

initialisation du dépôt

organisation

- pour utiliser Gitflow il faut deux branches permanentes: master et dev
- les branches feature, release et hotfixe sont dites éphémères (on les supprimera)
- les branches hotfixe et release sont mergées dans la branche master et dev
- la branche feature est mergée uniquement dans la branche dev
- les hotfixes sont taguées avec le deuxième chiffre Y de version X.Y, exemple tag 0.2
- les releases sont taguées avec le premier chiffre X de version X.Y, exemple tag 1.0
- si on a une release et que l'on fait une nouvelle feature => merge dans dev et feature
- Initialisation du projet

```
$ git init
# Exécute une commande sans effet --dry-run, liste les fichiers stagés
$ git add --all --dry-run > list.txt
$ rm list.txt
$ touch .gitignore
$ touch README.md
$ git commit -m "C1: ajout du projet"
$ git tag v1.0.0
```

- Création de la branche dev

```
$ git checkout -b dev
# Création d'un commit vide --allow-empty
$ git commit -m "C2: branche dev" --allow-empty
```

- envoie des deux branches, master et dev, créées sur le dépôt commits avec les réfs (option -u)

les refs envoient les pointeurs sur branche

```
# Option -u commits + réfs
$ git push -u --all

# Pour envoyer un/les tag(s)
```

```
git push origin v8

# ou plus simplement
git push origin --tags
```

création d'une feature

- Exemple de développement d'une feature, par Alan

```
$ git pull --all
# vérification de l'import des branches distantes
$ git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/dev
remotes/origin/master
```

création de la branche dev

attention, la branche dev n'est pas automatiquement créée, vous devez faire:

```
$ git checkout remotes/origin/dev
$ git checkout -b dev
```

- Création d'une feature à partir de la branch dev

```
$ git branch
master
*dev
$ git checkout -b feature_routes
# commit pour prévenir que l'on fait une nouvelle feature
$ git commit -m "feature: task route" --allow-empty

$ git push -u origin feature_routes
```

- Fin de la feature, Alan doit commiter

```
$ git pull --all
$ git add .
$ git commit -m "feature: task mise en place des routes app"
$ git push -u --all
```

- Revu de code par Antoine Si tout marche bien, Antoine va merger la nouvelle feature dans la branche dev

```
$ git pull --all
$ git merge --no-ff feature_routes -m "feature: task route terminée -> branch dev"
$ git push -u origin dev
```

- Suppression de la branche feature_routes par Alan

```
$ git pull --all
$ git branch -d feature_routes
# attention pour supprimer une branche distante bien mettre :
$ git push origin :feature_routes
```

- Création de la branche de version par Lead_Antoine, elle n'acceptera plus que des correctifs

```
$ git pull --all
$ git checkout dev
$ git checkout -b release-v1.0
$ git commit --allow-empty -m "Release-v1.0"
$ git push -u origin release-v1.0
```

- Intégration, par Lead_Antoine, de la release dans la branche master et dev

```
$ git pull --all
$ git checkout master
$ git merge --no-ff release-v1.0 -m "version master stable v1.0"
$ git push
# création du tag release
$ git tag v1.0
# publication des tags sur le serveur
$ git push --tags
$ git checkout master
$ git merge --no-ff release-v1.0 -m "version dev stable v1.0"
$ git push
# suppression de la branche release
$ git branch -d release-v1.0
$ git push origin :release-v1.0
```

correction d'un bug sur la branche master, hotfixe

correction du "bugname" par Antoine

```
$ git pull --all
$ git checkout master
```

```
$ git checkout -b hotfix_bugname
# fin de la correction du bug
$ git add --all
$ git commit -m "Fix: bugname"
# option -u pour publié toutes les refs
$ git push -u origin hotfix_bugname
```

Lead_Antoine fait la revu du code et intégre le correctif à la branche master

```
$ git checkout master
$ git pull --all
$ git checkout remotes/origin/hotfix_bugname
$ git checkout -b hotfix_bugname
$ git merge --no-ff hotfix_bugname
$ git push
# tag pour le correctif
$ git tag v1.0.1
$ git push --tags
# mettre à jour la branche dev
$ git checkout dev
$ git merge --no-ff hotfix_bugname
$ git push -u origin dev
# suppression de la branche hotfix
$ git branch -d hotfix_bugname
$ git push origin :hotfix_bugname
```