D'autres outils

Romain THERRAT

POCKOST

14 décembre 2020

Sommaire

Présentation

- Des règles simples
- Limite les conflits
- Favorise un historique propre
- Peut être comparé aux fonctionnalités de `pull requests`
- Permet de structurer son organisation

Deux branches pour les gouverner toutes

En suivant le principe git-flow nous n'avons que deux branches principales

- master : Version du code en production
- develop : Version du code en développement

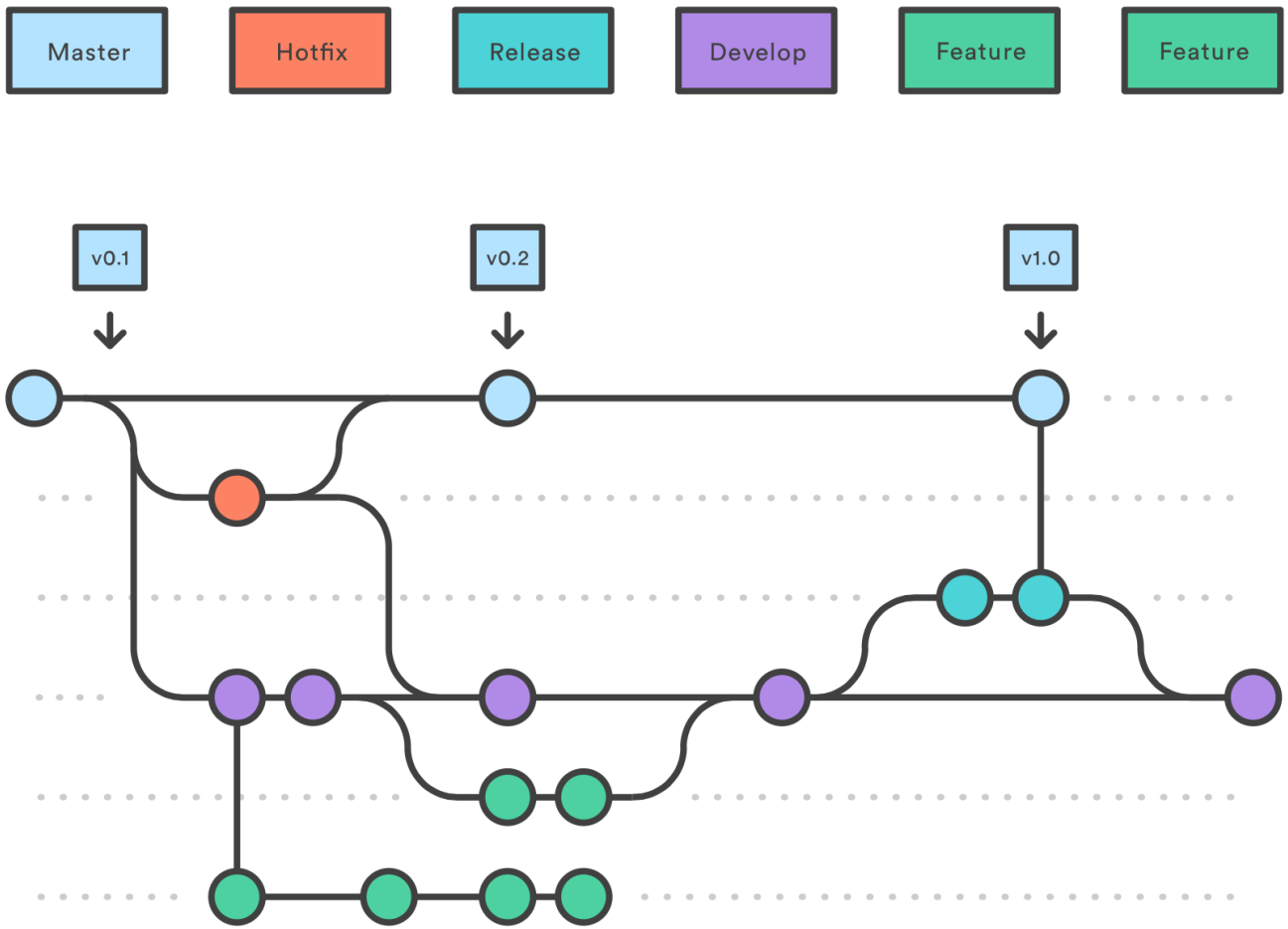
Ces deux branches sont protégées en écriture.

Les autres branches

Nous pouvons créer autant de branches que nous voulons suivant trois types différents

- feature : Travail sur une nouvelle fonctionnalité
- release : Préparation d'une nouvelle version (recette)
- hotfix : C'est le caca . . . je patche vite !

Fonctionnement



feature

Je dois travailler sur une nouvelle fonctionnalité.

```
$ git checkout develop
$ git checkout -b feature/my_feature
```

Quand j'ai fini

```
$ git checkout develop
$ git merge feature/my_feature --no-ff
$ git branch -d feature/my_feature
```

hotfix

Je dois corriger un bug en production

```
$ git checkout master
$ git checkout -b hostfix/my_bug
```

Une fois la correction faite

```
git checkout develop
git merge hotfix/my_bug --no-ff

git checkout master
git merge hotfix/my_bug --no-ff
git tag v1.42.1

git branch -d hotfix/my_bug
```

release

Je dois préparer une nouvelle version de l'application.

```
$ git checkout develop
$ git checkout -b release/v2.42
```

Je fais ensuite passer l'ensemble des tests sur le code de cette branche. Si tout se passe bien.

```
$ git checkout develop
$ git merge release/v2.42 --no-ff
$ git checkout master
$ git merge release/v2.42 --no-ff
$ git tag v2.42
$ git branch -d release/v2.42
```

La surcouche

C'est pas un peu compliqué tout ca ?

- Beaucoup de commandes
- Risque d'erreurs
- Une extension git
- ajoute les commandes `git flow`

Installation de git-flow

Désolé c'est écrit tout petit mais ca rentrait pas. . .

```
$ curl -OL \
  https://raw.github.com/nvie/gitflow/develop/contrib/gitflow-installer.sh
$ chmod +x gitflow-installer.sh
# ./gitflow-installer.sh
```

Configuration du projet

Il faut tout d'abord convertir notre dépôt au format git-flow.

```
$ git flow init

 Which branch should be used for bringing forth production
 releases?
   - cookie
   - master
Branch name for production releases: [master]
```

Démonstration

Utilisation

Créer une nouvelle branche

```
$ git flow feature start my_feature
$ git flow release start 1.42
$ git flow hotfix start 1.42.1
```

Merger mes modifications

```
$ git flow feature finish my_feature
$ git flow release finish 1.42
$ git flow hotfix finish 1.42.1
```

Démonstration

Démonstration

Sommaire

Labs

- Installer `git-flow`
- Convertir notre projet au format `git-flow`
- Créer une nouvelle `feature` nommée `change-title`
- Changer le titre d'`index.html` et `commiter` le résultat
- Terminer la `feature`
- Créer une nouvelle `release` nommé `1.42`
- Si tout vous semble bon terminer la `release`
- Penser à mettre un message pour le `tag`
- Créer un `hotfix` nommé `1.42.1`
- Modifier le titre de la page pour ajouter un smiley dedans
- Terminer votre `hotfix`