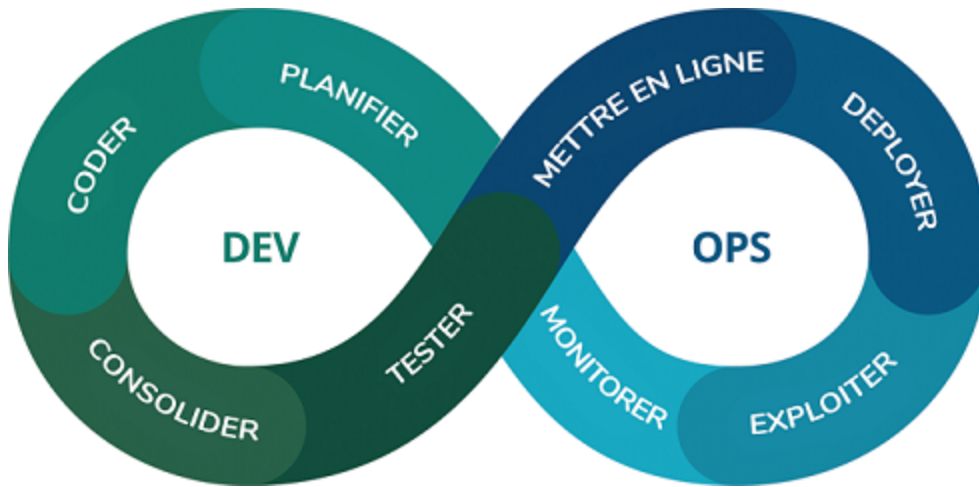


Intégration continue (CI) et déploiement continu (CD)



Source de l'illustration : nemesis-studio.com

Mise en pratique CI-CD sous Gitlab

par **Fabien Barbaud** - [@BarbaudFabien](https://twitter.com/BarbaudFabien)

Intégration continue

Continuous integration

L'**intégration continue** est un ensemble de pratiques utilisées en génie logiciel consistant à **vérifier** à chaque **modification de code source** que le résultat des modifications ne produit **pas de régression** dans l'application développée.

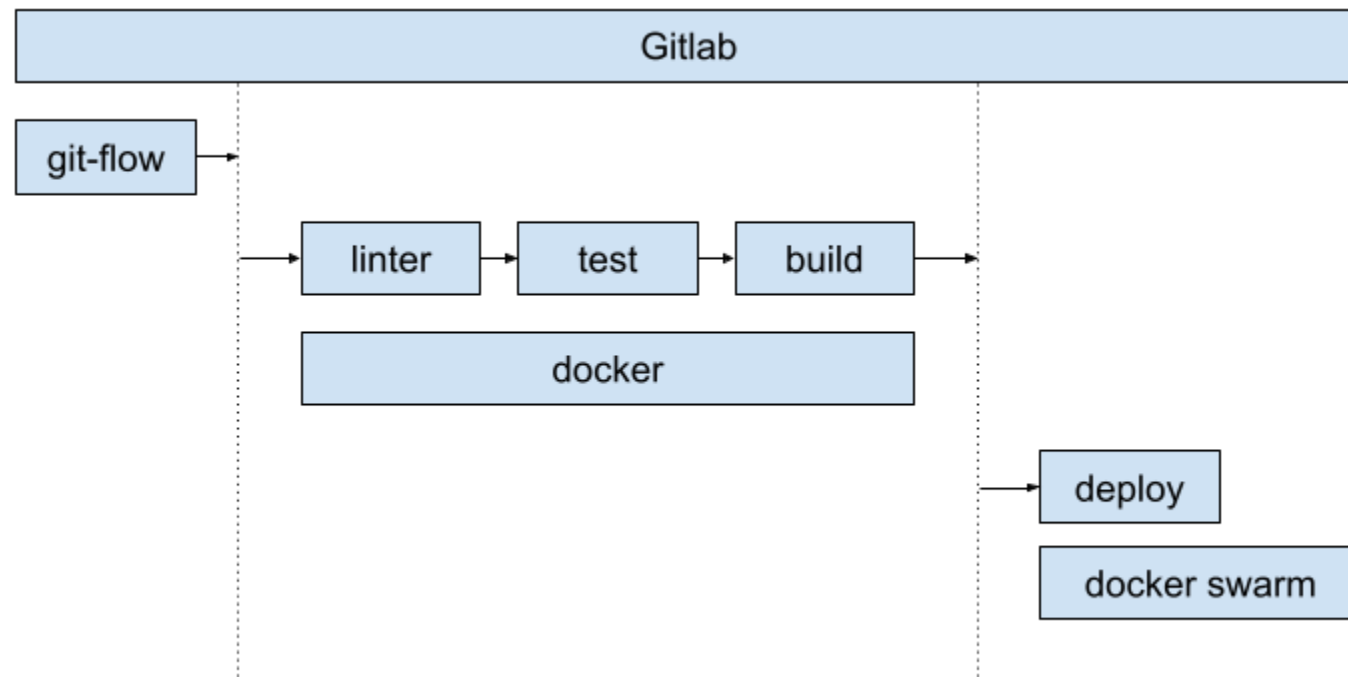
[Wikipedia](#)

Déploiement continu

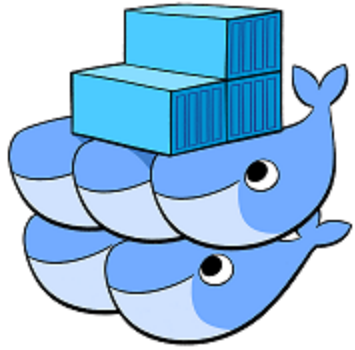
Le **déploiement continu** ou *Continuous deployment* (CD) en anglais, est une approche d'ingénierie logicielle dans laquelle les fonctionnalités logicielles sont **livrées fréquemment** par le biais de **déploiements automatisés**.

[Wikipedia](#)

Notre objectif :



Docker Swarm



Orchestrateur Docker

```
$ docker swarm init
```

Gitlab



GitLab

GitLab est un **logiciel libre** de **forge** basé sur git proposant les fonctionnalités de wiki, un système de suivi des bugs, **l'intégration continue et la livraison continue**.

[Wikipedia](#)

Gitlab

```
$ git clone https://github.com/fabienbarbaud/gitlab-docker.git  
$ cd gitlab-docker  
$ docker stack deploy --compose-file docker-compose.yml gitlab
```


```
$ docker service ls
```

<http://localhost>

u: root

p: MySuperSecretAndSecurePass0rd!

Gitlab

 You won't be able to pull or push repositories via SSH until you add an SSH key to your profile

Add SSH key

Don't show again

<http://localhost/-/profile/keys>

<http://localhost/help/ssh/README#generate-an-ssh-key-pair>

```
$ ssh git@localhost
```


Gitlab

- Créer un premier projet "test" - <http://localhost/root/test>
- Cloner ce projet sur votre poste
- Faire un premier commit
- Un push

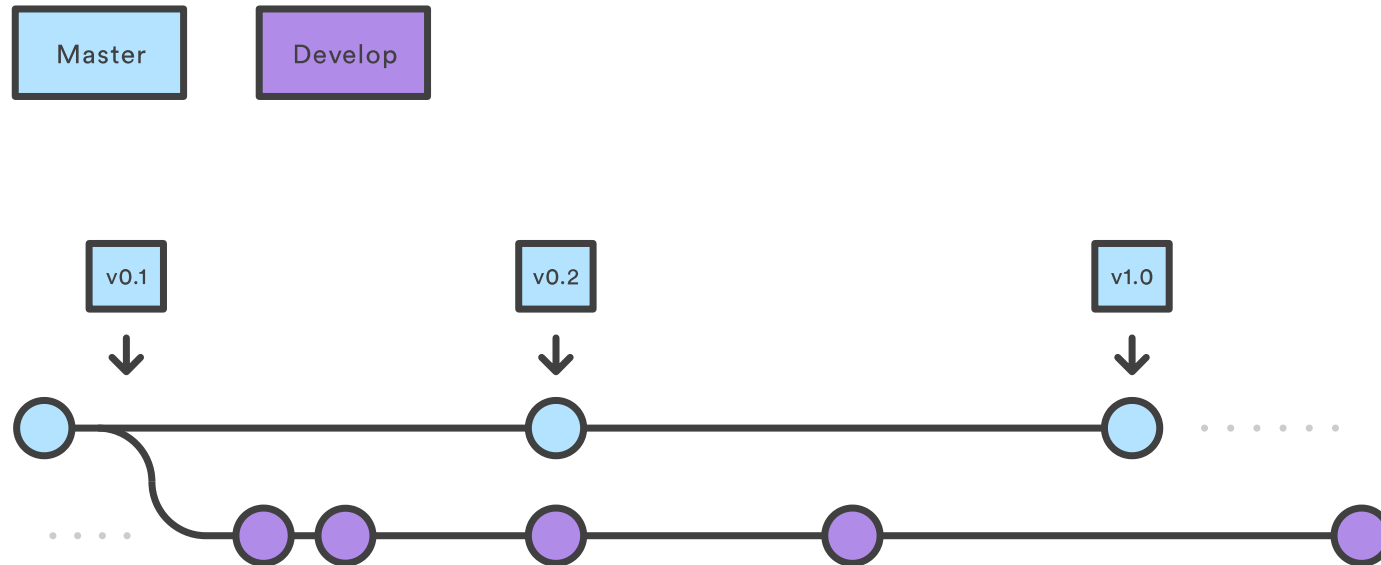
git-flow

Le workflow Gitflow définit **un modèle de création de branche strict** conçu autour de la **livraison de projet**. Cela fournit un **framework** solide pour la gestion de projets plus importants.

[Atlassian](#)

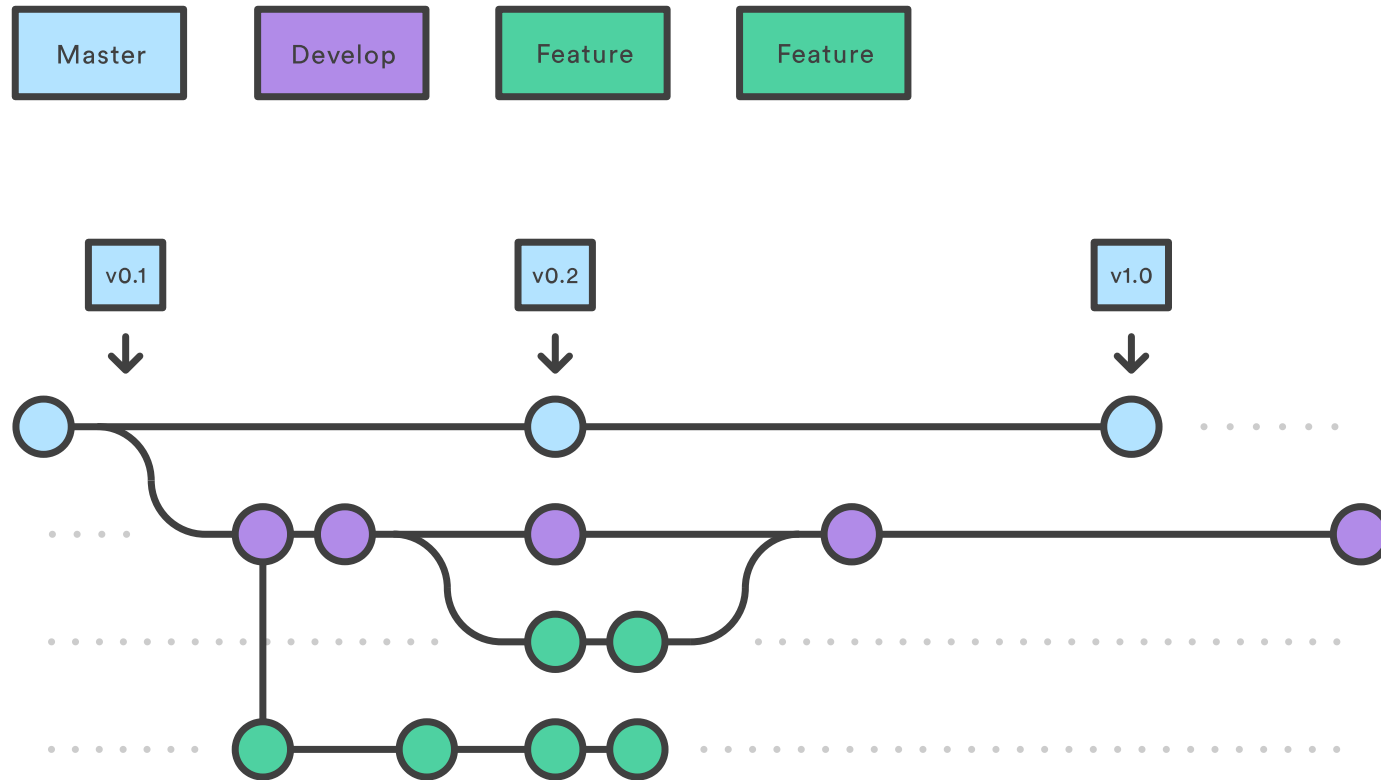
git-flow

Branches develop et master



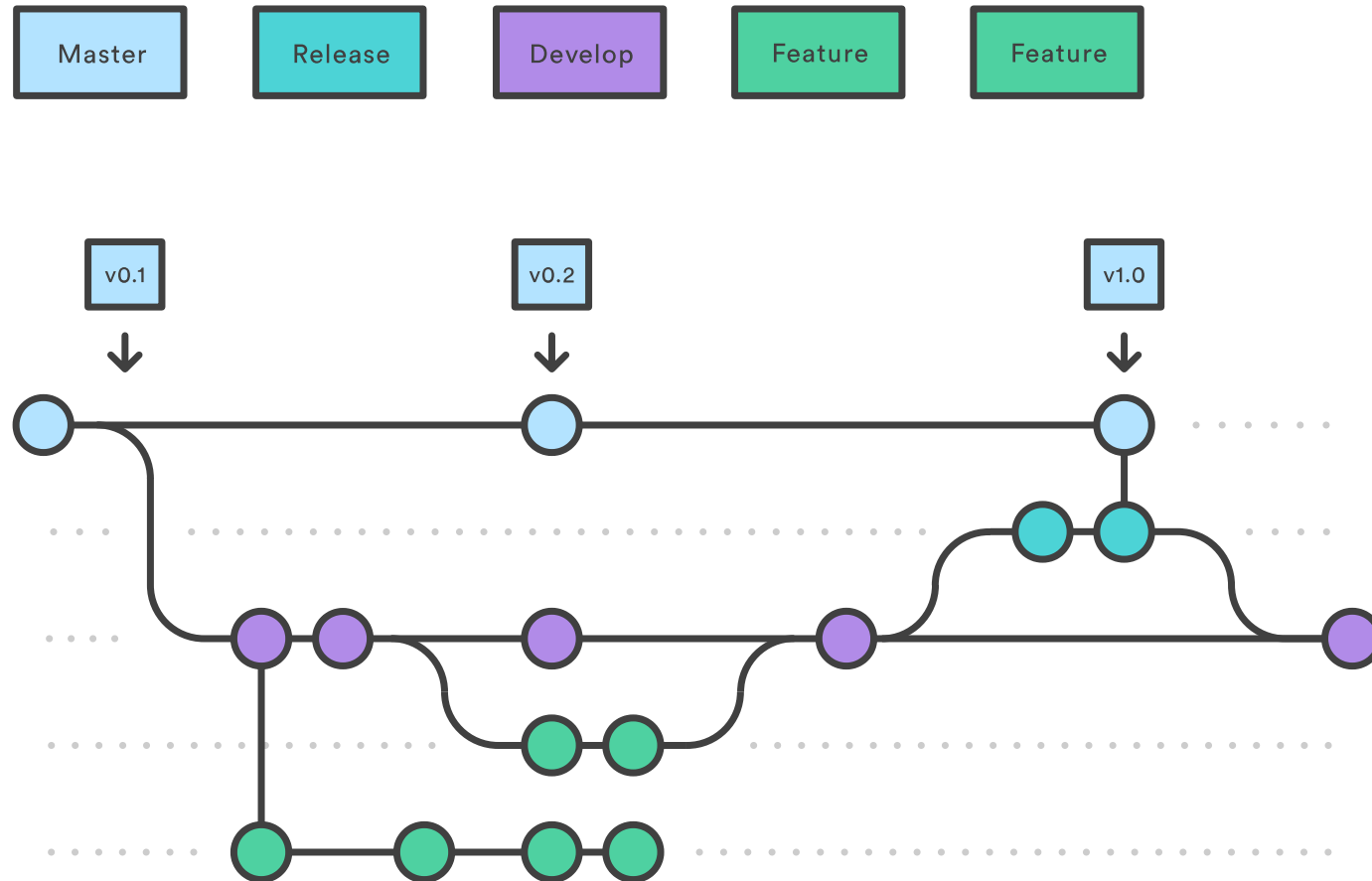
git-flow

Branche feature



git-flow

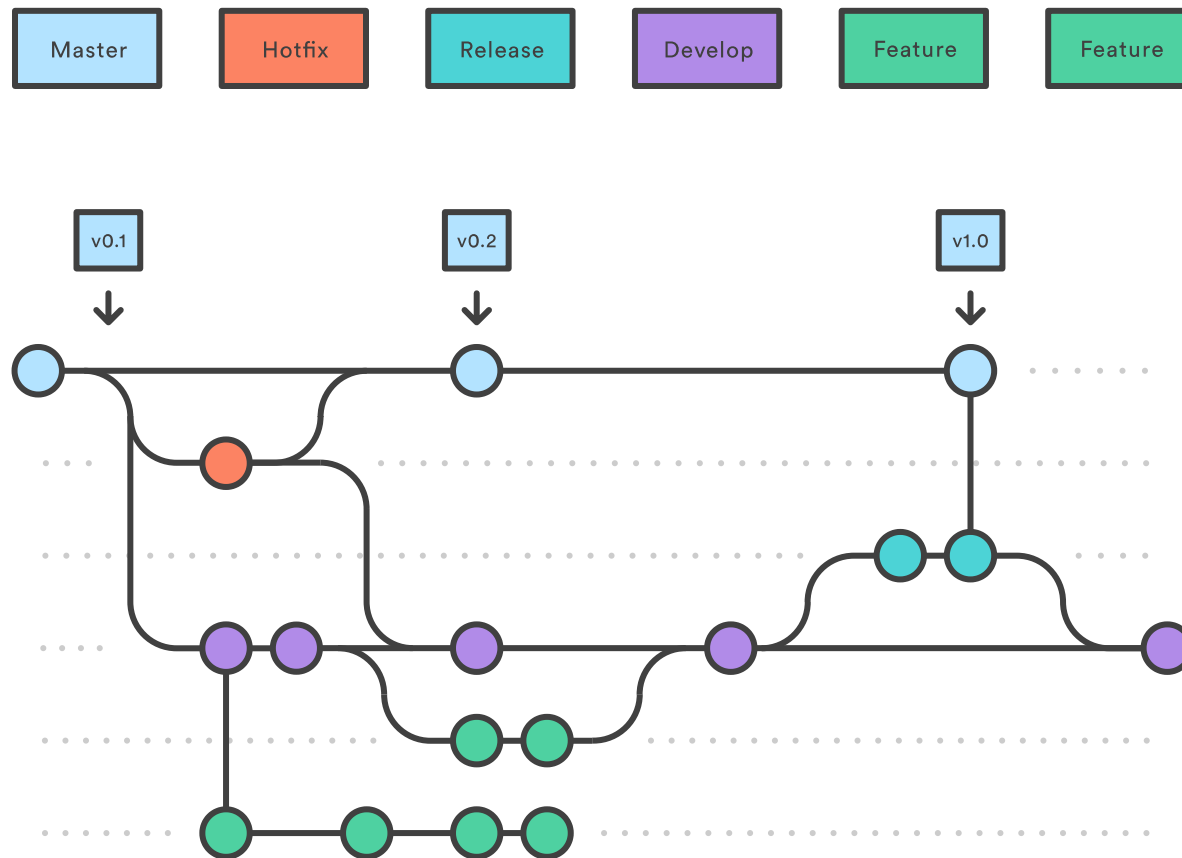
Branche release



Intégration continue (CI) et déploiement continu (CD)

git-flow

Branche hotfix



Intégration continue (CI) et déploiement continu (CD)

git-flow

Les commandes

```
$ git flow init  
$ git flow feature start ma-feature  
$ git flow finish -p  
$ git flow release start 0.1.0  
$ git flow finish -p  
$ git flow hotfix start 0.1.1  
$ git flow finish -p
```