Rapport programmation comparée

Devops : Gitlab CI vs. Github actions vs. Codefresh

NDIONE Mouhamed
HADJAB Hanane

DevOps, Qu’est ce que c’est ?

Le terme DevOps est composé de deux mots**“Dev”** représentant les équipes de développeurs et **“Ops”** représentant les équipes d’exploitation systèmes qui se chargent de mettre les applications en production, de les mettre à la disposition des utilisateurs et d'assurer leur maintenance.

DevOps permet de réunir le développement logiciel et l’administration de systèmes et d’architecture.

Intégration continue

L'intégration continue est une méthode de développement logiciel DevOps, ça permet d'exécuter d’une manière automatique les tests unitaires sur les nouveaux changements de codes pour détecter immédiatement n'importe quelle erreur, et ça automatise la génération et les tests de code.

Déploiement continue

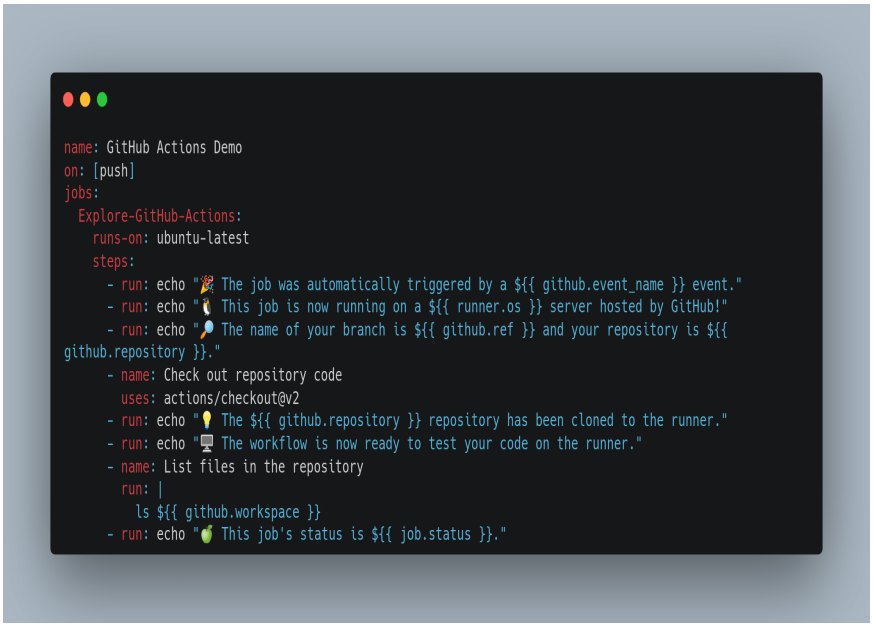
Le déploiement continu est une méthode de développement logiciel DevOps et la suite de l’intégration continue. Une fois que les tests sont validés sur l’environnement de dev, il faut les mettre en production.

GitHub - GitHub Actions

GitHub actions est une plateforme d’intégration continue et de déploiement continu, ça automatise tous les workflows logiciels comme les pipeline de déploiement, tests et les builds.

Il permet l’exécution des workflows lorsque des évènements se produisent.

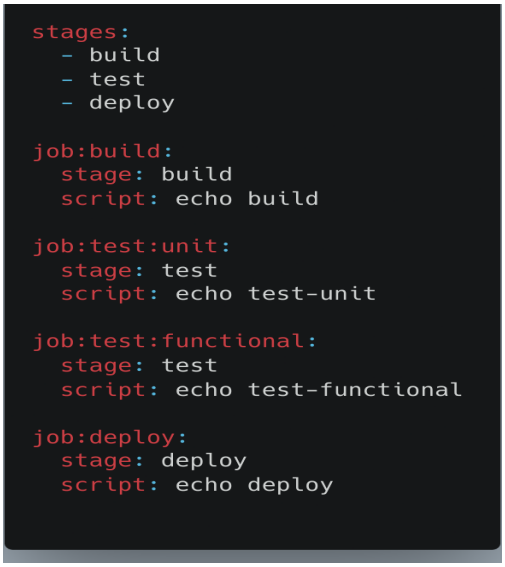
Un exemple de fichier .yml qui représente une pipeline qui sera exécuté au moment de push



```yaml
name: GitHub Actions Demo
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${{ github.event_name }} event."
      - run: echo "🐧 This job is now running on a ${{ runner.os }} server hosted by GitHub!"
      - run: echo "🔎 The name of your branch is ${{ github.ref }} and your repository is ${{
github.repository }}."
      - name: Check out repository code
        uses: actions/checkout@v2
      - run: echo "💡 The ${{ github.repository }} repository has been cloned to the runner."
      - run: echo "🖥️ The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${{ github.workspace }}
      - run: echo "🍏 This job's status is ${{ job.status }}."
```

Gitlab - Gitlab-CI

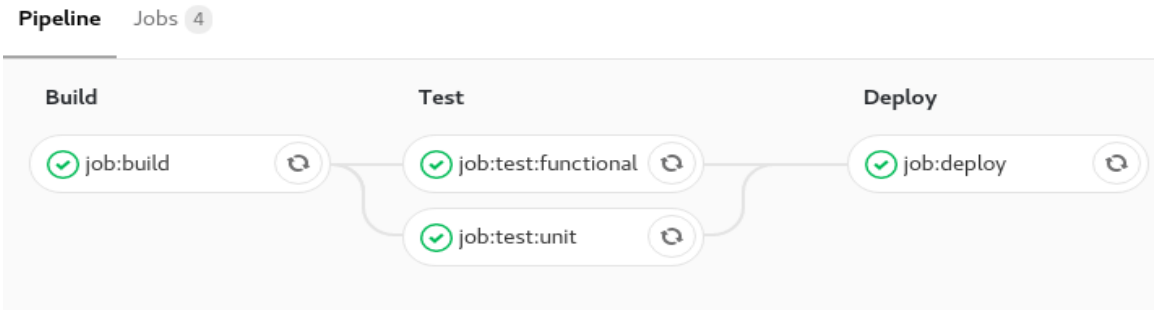
gitlab ci s'agit du module de CI/CD (Intégration continue et de déploiement continu) proposé par gitlab.
Il permet d'automatiser entre autres les builds, les tests, les livraisons et les déploiements.
Il dispose de nombreuses fonctionnalités et utilise la format de fichier yaml avec le fichier **.gitlab-ci.yml** qui permet de décrire le pipeline de votre projet.



```yaml
stages:
  - build
  - test
  - deploy

job:build:
  stage: build
  script: echo build

job:test:unit:
  stage: test
  script: echo test-unit

job:test:functional:
  stage: test
  script: echo test-functional

job:deploy:
  stage: deploy
  script: echo deploy
```

Ce schéma ci dessous représente un exemple de fichier **.gitlab-ci.yml** qui présente trois **étapes** qui sont builds, test et deploy. Les stages sont les étapes d'exécution et s'exécutent en séquentiel.

On a aussi les jobs qui contiennent les scripts qu'on souhaite exécuter à un stage défini. Les jobs qui sont dans le même stage s'exécutent en parallèle. Ci dessous, le schéma du pipeline résultant :



Codefresh

Codefresh est un outil d'intégration continue et de déploiement continu native de docker. Pour fonctionner, Codefresh a besoin d'accéder à un dépôt git afin d'être informé et de lancer le pipeline lorsqu'un événement comme un "push" survient. Ces dépôts git peuvent être acceder via github, gitlab ou bitbucket par exemple.
Ceci lui permet de regrouper dans un seul tableau de bord plusieurs projets de différents dépôts git et leurs pipelines.

Comparaison

Les trois outils de CI/CD présentés ci-dessus proposent pas mal de fonctionnalités similaires. Ils supportent tous **docker**, proposent une **API REST** pour la gestion externe, exécutent les **jobs en parallèle** et permettent aussi de configurer ses propres **runners.**
En revanche, ce qui fait la particularité de **Codefresh** est qu'elle permet **la mise en cache des images dockers** et l'exécution **des stages en parallèle** ce qui aura tendance à donner un temps d'exécution meilleur comparé à **github action** ou à **gitlab-ci**. De plus, pour un projet on peut avoir **plusieurs pipelines** avec Codeflesh, chose qu'on a pas avec github action et gitlab-CI. Le fait d'avoir plusieurs pipelines pour un projet permet d'avoir par exemple un pipeline par microservice.
Github actions et **codefresh** proposent un **marketplace** qui fournit pas mal de modules qui permettent de rapidement mettre en place son pipeline.

Conclusion

Un outil d'intégration continue et de déploiement continu est aujourd'hui presque indispensable pour mener à bien un projet de développement.
L'intégration continue et de déploiement continu permettent d'automatiser les builds, les tests et les déploiements tout en mettant un ensemble de règles que le code doit respecter.
Beaucoup d'outils existent et disposent de bonnes plaquettes marketing.

Ces trois outils qu'on vous a présenté proposent presque les mêmes fonctionnalités et à des différences minimes. La **migration** de l'un vers l'autre est possible dans plusieurs cas.
A côté des différences techniques et des différences tarifaires qui nous ont moins intéressé dans notre sujet, se pose d'autres questions: **l'outil est-il open-source ou pas** ? L**'outil respecte-t-il les licences sur nos projets ?**
Le rachat de **Github** par Microsoft a fait un tollé sur le web, la communauté du libre très déçue, certains ont migré vers **Gitlab**.
L'annonce de **Github copilot**, très bien perçue au début, a fait l'objet de beaucoup de critiques par la suite car l'outil a été entraîné sur des dépôts de code publiques dont la plupart sont **sous licence** et sous la **protection du droit d'auteur.**

Glossaires :

runner : machine sur lesquelles sont exécutés les pipelines

stage : compose les étapes du pipeline

steps : équivalent de **stage** avec github actions

job : défini les scripts à exécuter à un stage

Github Copilot : Intelligence artificiel proposé par github et permet l'autocompletion de code

CI/CD : Intégration continue / Déploiement continue

Références :

https://docs.github.com/en/actions

https://rigorousthemes.com/blog/gitlab-vs-github-which-is-better/

https://codefresh.io/

https://docs.gitlab.com/ee/ci/

https://rigorousthemes.com/blog/gitlab-vs-github-which-is-better/

https://codefresh.io/continuous-integration/codefresh-versus-gitlabci/

https://knapsackpro.com/ci_comparisons/codefresh-ci/vs/github-actions

https://analyticsindiamag.com/why-are-people-criticising-github-copilot/