

Intégration continue avec GitLab CI

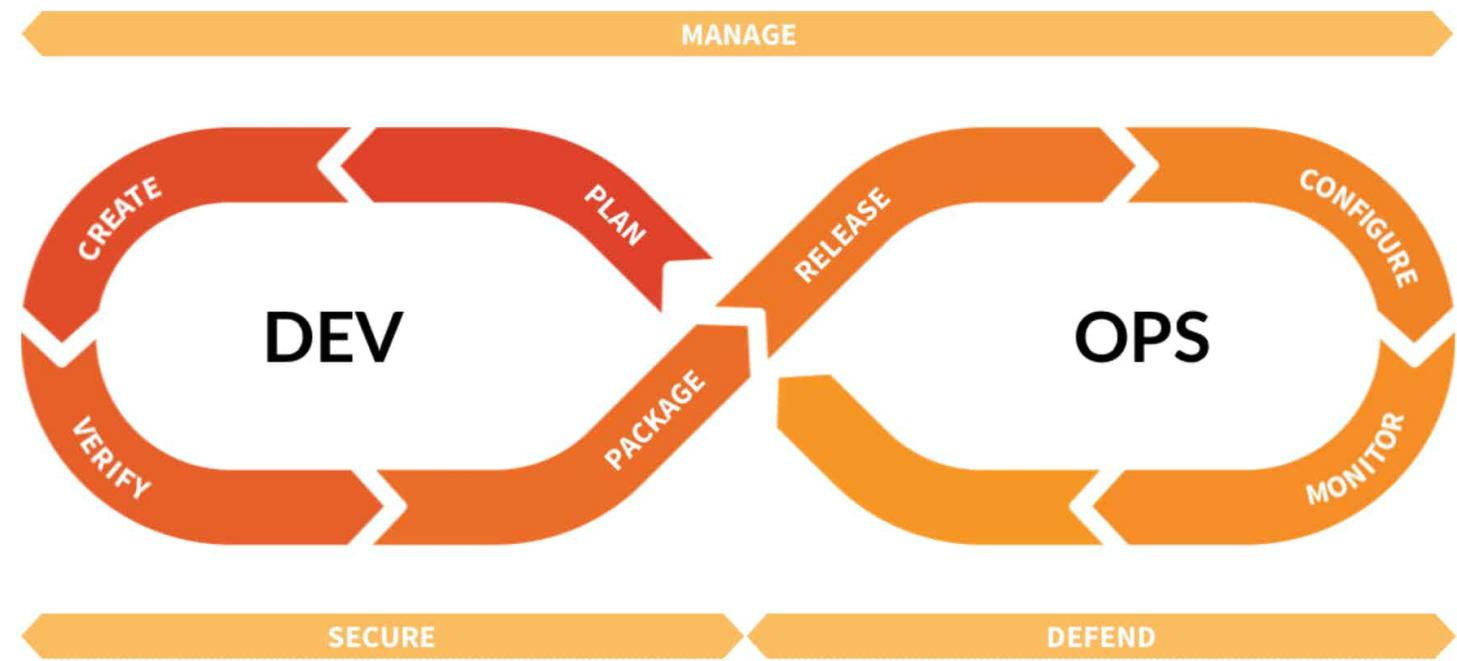


Intégration continue avec GitLab CI, F-E Goffinet, 2021

1

Téléchargements des supports

- PDF
- MOBI
- EPUB
- PPT



Intégration Continue
avec
GitLab CI



Intégration continue avec GitLab CI, F-E Goffinet, 2021

© François-Emmanuel Goffinet 2021

2

1. Introduction au projet GitLab



Intégration continue avec GitLab CI, F-E Goffinet, 2021

3

GitLab est un outil de gestion du cycle de vie de DevOps basé Web qui fournit un gestionnaire de référentiel Git fournissant des fonctionnalités wiki, de suivi des problèmes et de pipeline CI/CD. Il est développé sous licence open-source par GitLab Inc.



Le logiciel se décline en quatre produits :

- GitLab CE (Community Edition) - auto-hébergé et gratuit, support communautaire.
- GitLab EE (Enterprise Edition) - auto-hébergé et payant, fonctionnalités supplémentaires.
- GitLab.com - SaaS et gratuit.
- GitLab.io - Instance privée gérée par GitLab Inc.

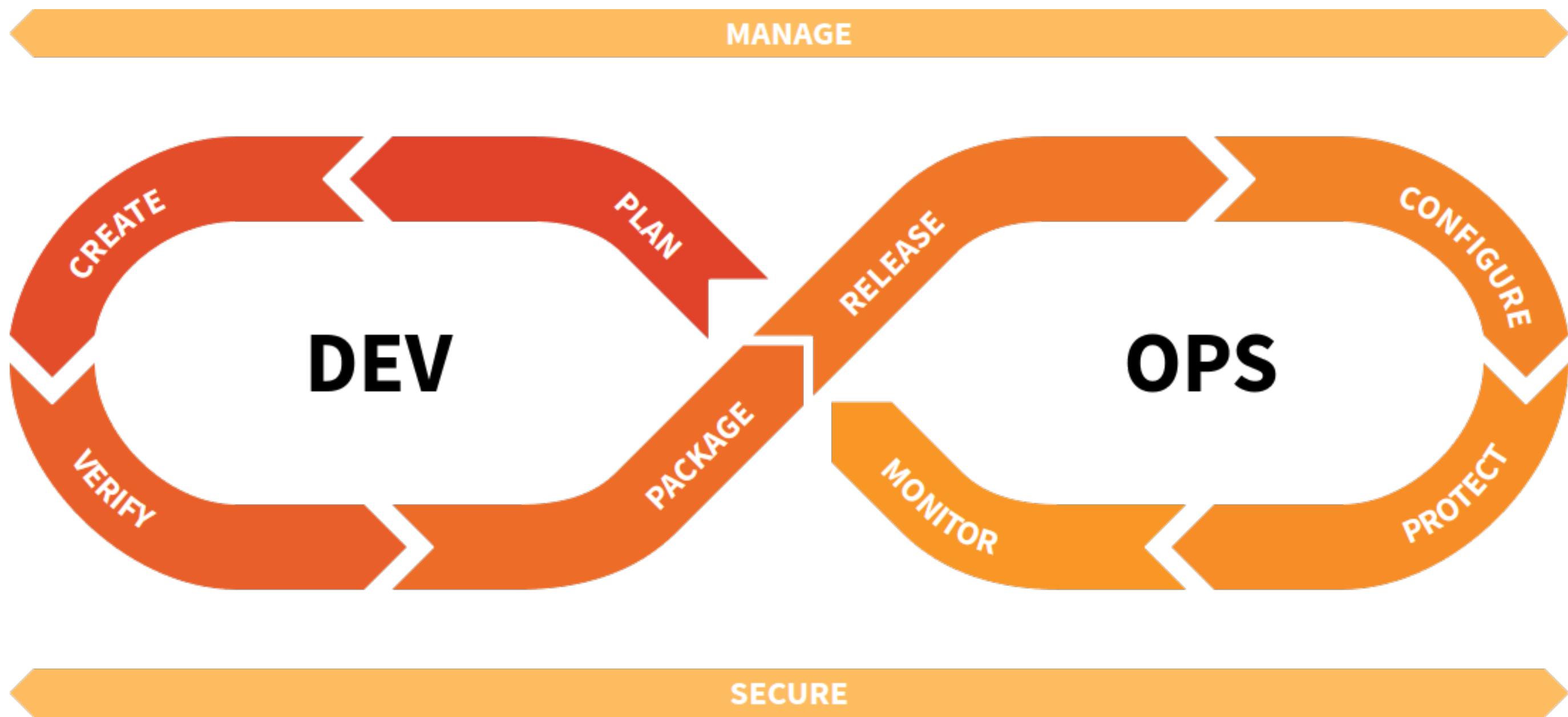
Les outils comparables sont par exemple [GitHub](#) ou [Bitbucket](#).

2. Introduction à DevOps avec GitLab CI



Intégration continue avec GitLab CI, F-E Goffinet, 2021

La documentation de GitLab CI sur trouve à l'adresse <https://docs.gitlab.com/ee/ci/README.html>.



Un cycle de vie DevOps se compose de différentes étapes en boucle : "Plan", "Create", "Verify", "Package", "Release", "Monitor". De manière transversale "Manage" et "Secure" s'intéressent à toutes les étapes du cycle.



| DevOps Stage | Description |
|--------------|--|
| Manage | Statistiques et fonctions d'analyse. |
| Plan | Planification et gestion de projet. |
| Create | Fonctions SCM (Source Code Management) |
| Verify | Tests, qualité du code et fonctions d'intégration continue. |
| Package | Registre des conteneurs Docker. |
| Release | Release et de livraison de l'application. |
| Configure | Outils de configuration d'applications et d'infrastructures. |
| Monitor | Fonctions de surveillance et de métrique des applications. |
| Secure | Fonctionnalités de sécurité. |



Informations de départ

[Get started with GitLab](#)



Intégration continue avec GitLab CI, F-E Goffinet, 2021

11

Organize

Create projects and groups.

- [Create a new project](#)
- [Create a new group](#)



Intégration continue avec GitLab CI, F-E Goffinet, 2021

12

Prioritize

Create issues, labels, milestones, cast your vote, and review issues.

- [Create an issue](#)
- [Assign labels to issues](#)
- [Use milestones as an overview of your project's tracker](#)
- [Use voting to express your like/dislike to issues and merge requests](#)

Collaborate

Create merge requests and review code.

- Fork a project and contribute to it
- Create a new merge request
- Automatically close issues from merge requests
- Automatically merge when pipeline succeeds
- Revert any commit
- Cherry-pick any commit



Test and Deploy

Use the built-in continuous integration in GitLab.

- [Get started with GitLab CI/CD](#)



Install and Update

Install and update your GitLab installation.

- [Install GitLab](#)
- [Update GitLab](#)
- [Explore Omnibus GitLab configuration options](#)



3. Projet de départ GitLab CI avec Pages



Intégration continue avec GitLab CI, F-E Goffinet, 2021

17

GitLab Pages est une fonctionnalité qui permet de publier des sites web statiques directement à partir d'un référentiel dans GitLab. La documentation de départ accessible à partir de cette page : [Creating and Tweaking GitLab CI/CD for GitLab Pages.](#)



Un cycle d'intégration continue dans Gitlab CI est défini à partir d'un fichier de configuration écrit en YAML. Le fichier est placé à la racine du projet sous le nom réservé de `.gitlab-ci.yml`.

Un "pipeline" est une suite de "stages", soit un flux d'étapes. Un "stage" exécute des jobs. Ceux-ci sont définis par des variables, des commandes et la génération d'"artifacts". Un "artifacts" est le résultats d'une exécution gardé en mémoire pour traitement dans le "pipeline".

L'exécution des jobs sont réalisées dans des conteneurs Docker sur n'importe quel machine ou Pod K8s (Kubernetes) enregistrés comme "Gitlab Runner".

[GitLab CI/CD Pipeline Configuration Reference](#)



Un "job" spécial nommé "pages" génère tous les "artifacts" d'un site web dans le dossier spécial public.

Job spécial Pages et dossier public /



Essai local avec un exemple Gitlab

Référentiel à importer : [Example GitBook site using GitLab Pages](https://gitlab.com/pages/gitbook.git)

```
yum -y install git
```

```
git clone https://gitlab.com/pages/gitbook.git
```

```
cd gitbook
```

```
ls -la
```

```
docker run -it -p 4000:4000 -v $PWD:/gitbook node:latest bash  
cd /gitbook  
npm install gitbook-cli -g  
gitbook install  
gitbook serve
```



Pipeline GitLab CI

Fichier .gitlab-ci.yml



Intégration continue avec GitLab CI, F-E Goffinet, 2021

25

```
# requiring the environment of NodeJS 10
image: node:10

# add 'node_modules' to cache for speeding up builds
cache:
  paths:
    - node_modules/ # Node modules and dependencies

before_script:
  - npm install gitbook-cli -g # install gitbook
  - gitbook fetch 3.2.3 # fetch final stable version
  - gitbook install # add any requested plugins in book.json

test:
  stage: test
  script:
    - gitbook build . public # build to public path
  only:
    - branches # this job will affect every branch except 'master'
  except:
    - master

# the 'pages' job will deploy and build your site to the 'public' path
pages:
  stage: deploy
  script:
    - gitbook build . public # build to public path
  artifacts:
    paths:
      - public
  expire_in: 1 week
  only:
    - master # this job will affect only the 'master' branch
```



4. CI/CD Gitbook



Intégration continue avec GitLab CI, F-E Goffinet, 2021

27

Pipeline GitLab CI

goffinet > gitbook-publication > Pipelines > #55652670

passed Pipeline #55652670 triggered 3 minutes ago by goffinet

Référentiel à importer : [Gitbook Publication](#)

Update SUMMARY.md

8 jobs from master in 2 minutes and 29 seconds

latest

eef1887e ...

Pipeline Jobs 8



Intégration continue avec GitLab CI, F-E Goffinet, 2021

28

Fichier gitlab-ci.yml :

```
# This pipeline run three stages Test, Build and Deploy  
stages:  
  - test  
  - build  
  - deploy
```

```
image: goffinet/gitbook:latest
```

```
# the 'gitbook' job will test the gitbook tools
gitbook:
  stage: test
  image: registry.gitlab.com/goffinet/gitbook-gitlab:latest
  script:
    - 'echo "node version: $(node -v)"'
    - gitbook -V
    - calibre --version
  allow_failure: false
```

```
# the 'lint' job will test the markdown syntax
lint:
  stage: test
  script:
    - 'echo "node version: $(node -v)"'
    - echo "markdownlint version:" $(markdownlint -V)
    - markdownlint --config ./markdownlint.json README.md
    - markdownlint --config ./markdownlint.json *.md
  allow_failure: true
```

```
# the 'html' job will build your document in html format
html:
  stage: build
  dependencies:
    - gitbook
    - lint
  script:
    - gitbook install # add any requested plugins in book.json
    - gitbook build . book # html build
  artifacts:
    paths:
      - book
  expire_in: 1 day
only:
  - master # this job will affect only the 'master' branch the 'html' job will build your document in pdf format
allow_failure: false
```

```
# the 'pdf' job will build your document in pdf format
pdf:
  stage: build
  dependencies:
    - gitbook
    - lint
  before_script:
    - mkdir ebooks
  script:
    - gitbook install # add any requested plugins in book.json
    - gitbook pdf . ebooks/${CI_PROJECT_NAME}.pdf # pdf build
  artifacts:
    paths:
      - ebooks/${CI_PROJECT_NAME}.pdf
    expire_in: 1 day
  only:
    - master # this job will affect only the 'master' branch the 'pdf' job will build your document in pdf format
```

```
# the 'epub' job will build your document in epub format
epub:
  stage: build
  dependencies:
    - gitbook
    - lint
  before_script:
    - mkdir ebooks
  script:
    - gitbook install # add any requested plugins in book.json
    - gitbook epub . ebooks/${CI_PROJECT_NAME}.epub # epub build
  artifacts:
    paths:
      - ebooks/${CI_PROJECT_NAME}.epub
  expire_in: 1 day
only:
  - master # this job will affect only the 'master' branch
```

```
# the 'mobi' job will build your document in mobi format
mobi:
  stage: build
  dependencies:
    - gitbook
    - lint
  before_script:
    - mkdir ebooks
  script:
    - gitbook install # add any requested plugins in book.json
    - gitbook mobi . ebooks/${CI_PROJECT_NAME}.mobi # mobi build
  artifacts:
    paths:
      - ebooks/${CI_PROJECT_NAME}.mobi
  expire_in: 1 day
only:
  - master # this job will affect only the 'master' branch
```

```
# the 'pages' job will deploy your site to your gitlab pages service
pages:
  stage: deploy
  dependencies:
    - html
    - pdf
    - mobi
    - epub # We want to specify dependencies in an explicit way, to avoid confusion if there are different build jobs
  script:
    - mkdir .public
    - cp -r book/* .public
    - cp -r ebooks/* .public
    - mv .public public
  artifacts:
    paths:
      - public
only:
  - master
```

Déploiement sur Netlify

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

37

5. CI/CD Jekyll



Intégration continue avec GitLab CI, F-E Goffinet, 2021

38

Pipeline GitLab CI

Référentiel à importer : [Jekyll good-clean-read](#)

Fichier gitlab-ci.yml :

```
image: ruby:2.3

variables:
  JEKYLL_ENV: production
  LC_ALL: C.UTF-8

before_script:
  - bundle install

pages:
  stage: deploy
  script:
    - bundle exec jekyll build -d public
  artifacts:
    paths:
      - public
  only:
    - gitlab
```



6. CI/CD Mkdocs



Intégration continue avec GitLab CI, F-E Goffinet, 2021

40

Pipeline GitLab CI

Référentiel à importer : [mkdocs-material-boilerplate](#)

Fichier gitlab-ci.yml :

```
image: python:3.6-alpine

before_script:
  - pip install --upgrade pip && pip install -r requirements.txt

pages:
  script:
    - mkdocs build
    - mv site public
artifacts:
  paths:
    - public
only:
  - master
```



Déploiement sur Netlify

[Deployer sur Netlify](#)



Intégration continue avec GitLab CI, F-E Goffinet, 2021

42

7. CI/CD Maven - Apache Tomcat



Intégration continue avec GitLab CI, F-E Goffinet, 2021

43

Artifactory and Gitlab



Intégration continue avec GitLab CI, F-E Goffinet, 2021

44

Premier exemple

Exemple CI/CD avec Maven, lecture de l'exemple et application selon le document [Maven in five minutes](#).

Créer un dépôt sur Gilab et le cloner localement.

Importer une clé SSH.

Image Docker maven.

Pipeline :

- test
- build

Essai local

```
mvn archetype:generate -DgroupId=com.mycompany.app \
-DartifactId=my-app \
-DarchetypeArtifactId=maven-archetype-quickstart \
-DarchetypeVersion=1.4 -DinteractiveMode=false
cd my-app
docker run -it -v $PWD/my-app:/my-app maven bash
exit
```

Pipeline GitLab CI

Fichier .gitlab-ci.yml

```
image: maven:latest

build:
  stage: build
  script:
    - mvn package
  artifacts:
    paths:
      - target

test:
  stage: test
  script:
    - java -cp target/my-app-1.0-SNAPSHOT.jar com.mycompany.app.App
```



Initialisation d'un repo gitlab

```
git init
git add *
echo "target" >> .gitignore
git add .gitignore
git remote add origin https://gitlab.com/account/project.git
git push -u origin master
```



Second exemple

Cette fois ci avec l'archétype Maven "Webapp" et une phase/job "deploy"

- test
- build
- deploy



Déploiement sur Tomcat

...

Méthodes

SSH et Bash

SCP

Text Manager avec curl

Authentification

clé secrète

clé secrète

login/mot de passe



Intégration continue avec GitLab CI, F-E Goffinet, 2021

52

Variables cachées

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

53

Gitlab Runner

Exécution sur un Gitlab-Runner qui héberge le serveur applicatif.

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

54

Avertissement Slack

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

55

Pipeline GitLab CI

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

56

8. PHP projects



Intégration continue avec GitLab CI, F-E Goffinet, 2021

57

Test PHP projects using the Docker executor

PHP (PHP.gitlab-ci.yml)



Intégration continue avec GitLab CI, F-E Goffinet, 2021

58

```
# Select image from https://hub.docker.com/_/php/
image: php:latest

# Select what we should cache between builds
cache:
  paths:
    - vendor/

before_script:
  - apt-get update -yqq
  - apt-get install -yqq git ... libpcre3-dev libtidy-dev
  # Install PHP extensions
  - docker-php-ext-install mbstring mcrypt pdo_pgsql curl json intl gd xml zip bz2 opcache
  # Install & enable Xdebug for code coverage reports
  - pecl install xdebug
  - docker-php-ext-enable xdebug
  # Install and run Composer
  - curl -sS https://getcomposer.org/installer | php
  - php composer.phar install
```

```
# Bring in any services we need http://docs.gitlab.com/ee/ci/docker/using_docker_images.html#what-is-a-service
# See http://docs.gitlab.com/ee/ci/services/README.html for examples.
services:
  - mysql:5.7

# Set any variables we need
variables:
  # Configure mysql environment variables (https://hub.docker.com/r/_/mysql/)
  MYSQL_DATABASE: mysql_database
  MYSQL_ROOT_PASSWORD: mysql_strong_password

# Run our tests
# If Xdebug was installed you can generate a coverage report and see code coverage metrics.
test:
  script:
    - vendor/bin/phpunit --configuration phpunit.xml --coverage-text --colors=never
```

9. Installation d'un serveur GitLab CE



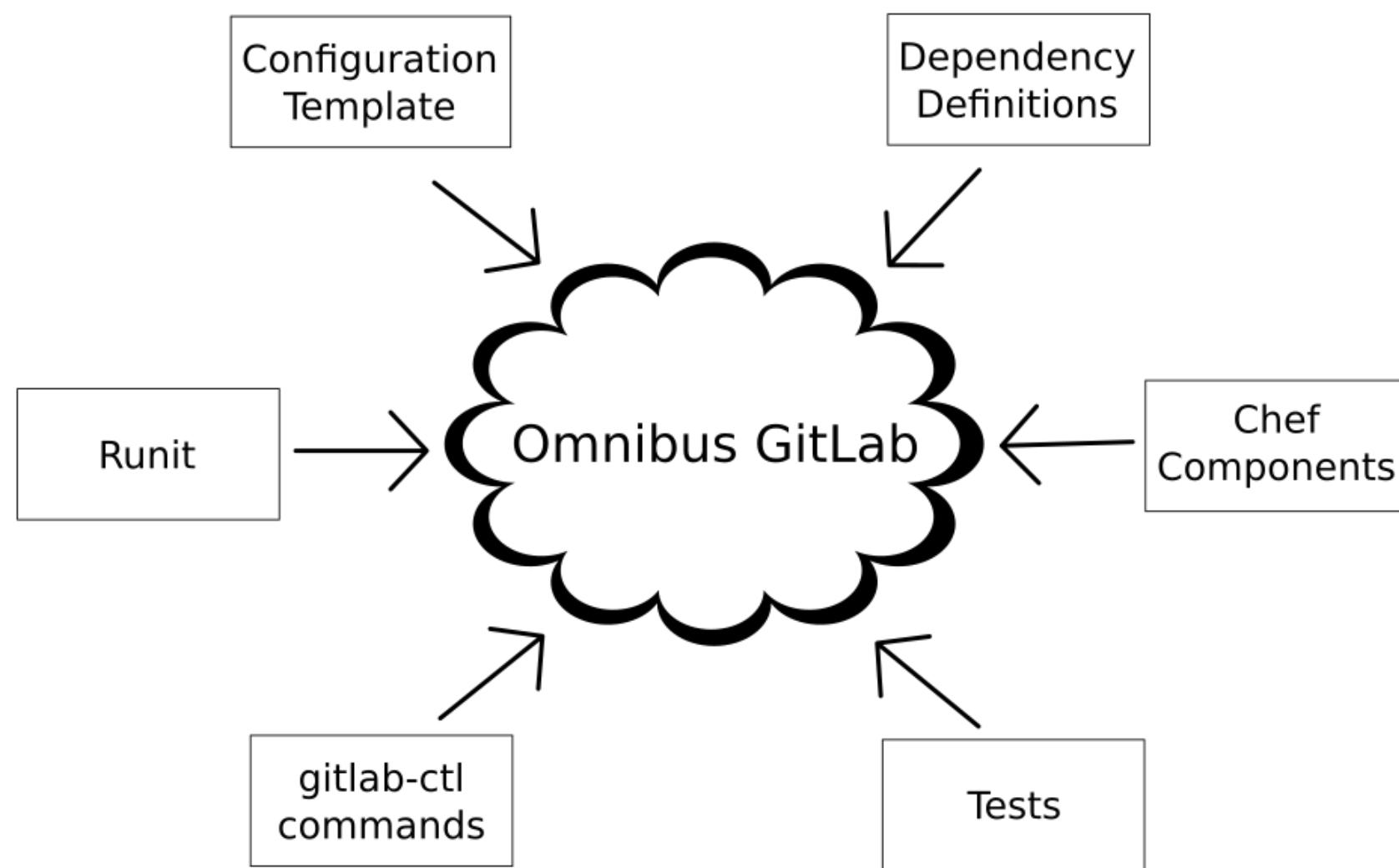
Intégration continue avec GitLab CI, F-E Goffinet, 2021

61

Omibus

Omnibus GitLab est une fourchette personnalisée du projet Omnibus de Chef, et il utilise des composants de Chef comme les cookbooks et les recipes pour exécuter la tâche de configuration de GitLab dans l'ordinateur d'un utilisateur. Le dépôt Omnibus GitLab sur GitLab.com héberge tous les composants nécessaires de l'Omnibus GitLab. Cela comprend les parties de l'Omnibus qui sont nécessaires pour construire le paquet, comme les configurations et les métadonnées du projet, et les composants liés au Chef qui seront utilisés dans l'ordinateur d'un utilisateur après l'installation.

Source de l'image : [Omnibus GitLab Architecture and Components](#)



Installation par Omnibus

Nous choisissons [une installation par Omnibus](#).

La variable `#{EXTERNAL_URL}` décide l'usage de HTTP ou HTTPS.

On installe un paquet rpm sur CentOS 7.

```
#tls=no
mkdir tmp ; cd tmp
export PUBLIC_IPV4=$(curl -s ipindo.io/ip)
export DNSDOMAIN="gitlab.${PUBLIC_IPV4}.nip.io"
if [[ "$tls" == "no" ]]; then
export EXTERNAL_URL="http://${DNSDOMAIN}"
openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -days 1000 -nodes -subj '/CN=${DNSDOMAIN}'
else
export EXTERNAL_URL="https://${DNSDOMAIN}"
fi
```



```
export GITLAB_VERSION="13.7.1"
yum -y install policycoreutils-python
yum -y install wget
repo="https://packages.gitlab.com/gitlab/gitlab-ce/packages/el/7"
wget --content-disposition ${repo}/gitlab-ce-${GITLAB_VERSION}-ce.0.el7.x86_64.rpm/download.rpm
export LC_ALL="en_US.UTF-8"
echo 'export LC_ALL="en_US.UTF-8"' >> .bashrc
rpm -Uvh gitlab-ce-${GITLAB_VERSION}-ce.0.el7.x86_64.rpm
```

A la première connexion sur l'interface Web, vous décidez d'un mot de passe `root`. Par défaut, les utilisateurs peuvent s'auto-enregistrer. Un compte comme `root` admet ces nouvelles demandes et octroie les droits aux utilisateurs. Ce comportement peut facilement être diminué ou être augmenté en terme de sécurité.

En cas de problème : [How to reset your root password](#)

Mise-à-jour

La mise à jour consiste à installer un nouveau packet dans la nouvelle version.

```
export GITLAB_VERSION="13.7.2"
yum -y install policycoreutils-python
yum -y install wget
repo="https://packages.gitlab.com/gitlab/gitlab-ce/packages/el/7"
wget --content-disposition ${repo}/gitlab-ce-${GITLAB_VERSION}-ce.0.el7.x86_64.rpm/download.rpm
export LC_ALL="en_US.UTF-8"
echo 'export LC_ALL="en_US.UTF-8"' >> .bashrc
rpm -Uvh gitlab-ce-${GITLAB_VERSION}-ce.0.el7.x86_64.rpm
```

Post-installation

- Mail : installation postfix 'Internet Site' ou [configure an external SMTP server](#)
- [LDAP](#)
- [Stocker les données dans des autres emplacements](#) ou encore sur [Stackoverflow](#)
- [Démarrer les services après les points de montage](#)
- [Les emplacements des données](#)
- [Configurer la timezone](#)
- [Modifier le logo](#)

Modèle AWS CloudFormation

...



Intégration continue avec GitLab CI, F-E Goffinet, 2021

69

10. Administration d'un serveur GitLab



Intégration continue avec GitLab CI, F-E Goffinet, 2021

70

Tâches de maintenance



Intégration continue avec GitLab CI, F-E Goffinet, 2021

71

Emplacement des fichiers

- L'emplacement par défaut des données : /var/opt/gitlab/git-data
- Fichier de configuration du serveur : /etc/gitlab/gitlab.rb

Gestion du serveur

- Reconfiguration du serveur : `gitlab-ctl reconfigure`
- Statut du serveur : `gitlab-ctl status`
- Tail sur les logs : `gitlab-ctl tail`
- Démarrer, arrêter ou redémarrer le serveur ou des services :
`gitlab-ctl start|stop|restart`

Backups

- Backup des configs et des datas