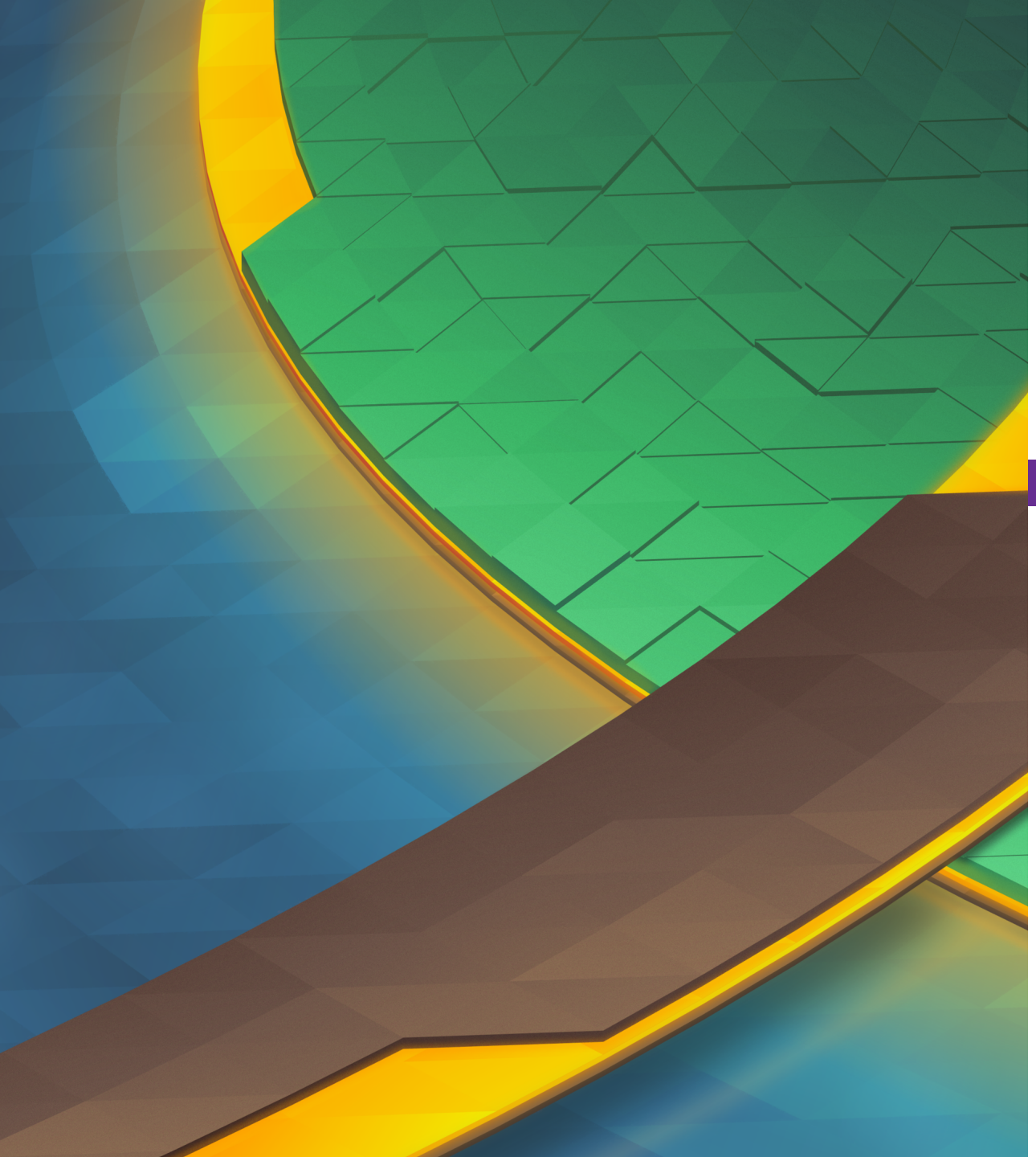


Gitlab et Gitlab-CI

Glenn Y. Rolland
<teaching@glenux.net>

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Préambule

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Objectifs de la formation

- Maîtriser les fondamentaux de GitLab comme un système de gestion de versions et de collaboration ;
- Structurer et organiser vos projets pour favoriser la réutilisabilité et la maintenance à long terme ;
- Collaborer efficacement au sein d'une équipe avec les outils de tracking et de revue de code intégrés ;
- Comprendre et implémenter des pipelines de GitLab-CI pour le CI/CD ;
- Intégrer et orchestrer des infrastructures complexes en utilisant les capacités de déploiement automatique de GitLab-CI ;
- Créer, tester et déployer des applications de manière agile et sécurisée ;
- Apprendre à monitorer et optimiser vos pipelines de CI/CD pour garantir des performances optimales et une livraison rapide.

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Qui êtes vous ?

Petit tour de présentation... avec 3 questions (1 minute chacun)

Passé

Quel est votre "bagage" ?

Quelle est votre expérience ?

Quelles sont vos compétences ?

Présent

Quelles circonstances vous amènent ici ?

Pourquoi participez vous à cette formation aujourd'hui ?

En quoi cette formation est importante pour votre équipe ?

Futur

Quel est votre objectif avec ce cours ?

Comment utiliserez-vous ces nouvelles compétences d'ici ?

Et d'ici 2 ans ?.. 5 ans ?

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Qui suis-je ?

2021 → aujourd'hui	**Auteur, conférencier et co-fondateur de CRYPTO-CHEMISTS** Formation et conseil sur l'impact des Blockchain & des technologies P2P.
2018 → aujourd'hui	**Directeur technique et co-fondateur de BOLDCODE** Développement et audit logiciel, web et mobile, offshoring éthique au Népal.
2017 → 2022	**Directeur technique et co-fondateur de DATA-TRANSITION** Gestion éthique des données, audit des SI, conformité au RGPD.
2010 → 2017	**Gérant et co-fondateur de NETCAT (GNUSIDE)** Infrastructures & systèmes en réseau, optimisation de la fiabilité, de la sécurité et de la performance.
2006 → 2010	**Ingénieur de recherche chez BEWAN (Pace Group)** Conception de systemes embarqués, et automatisation de la qualité logicielle.

Glenn ROLLAND © 2017-2022 - Reproduction interdite

La formation chez Orsys - en quelques mots

- 45 ans d'existance
- 7700 entreprises et administrations clientes
- 85000 personnes formées par an
- 1700 intervenants experts
- 97.5% de clients satisfaits
- 27 centres de formation en France
- 5000+ sessions intra-entreprise
- 7000+ sessions inter-entreprise
- 72,2M€ de chiffres d'affaires

Note: Chiffres de 2019

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Déroulement de la formation & règles du jeu

Horaires

- 9h00 - 12h30
- 13h30 - 17h00
- Des pauses le matin et l'après midi

Le cadre

- Liberté de parole dans le respect des autres et des objectifs de la formation
- Bienveillance, nous sommes dans un espace d’apprentissage
- Confidentialité de l’animateur et des participants sur les échanges

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Gitlab

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Rappels Git

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Trois états du dépot

- Work directory
- Stage
- History
  - Remote tracking branch (locale)
  - Remote branch

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Commandes Git de base

Configuration

```
$ git config
```

Initialisation

```
git init
```

```
git clone
```

Travail local

```
git add
```

```
git commit
```

Information

```
git status
```

```
git diff
```

```
git blame
```

Collaboration

```
git fetch
```

```
git pull
```

```
git push
```

Gestion des branches

```
git checkout
```

```
git branch
```

```
git merge
```

Gestion des snapshots

```
git rebase
```

```
git cherry-pick
```

```
git bissect
```

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Workflows Git

Workflow à base de branches

- Travail local ou distant
- Répartition du travail dans des branches distinctes
  - Pour la collaboration
  - Pour la gestion de la qualité
  - etc.
- Fusion locale
- Avec ou sans outils
  - ex: `git flow`

Workflow de fork

- Copie du repository
- Demande d'intégration des changements
  - `git send-email`
  - Merge/Pull request

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Références

- git - petit guide

Glenn ROLLAND © 2017-2022 - Reproduction interdite

À propos de Gitlab

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Qu’est ce que GitLab ?

- Logiciel libre
- Fournissant
  - un hébergement de dépôts Git
  - des outils pour gérer le déôt
  - des outils pour gérer les projets et la collaboration
- Développé initialement en Ruby / Rails
  - Réécrit en partie en Go par la suite
  - Frontend en VueJS

Glenn ROLLAND © 2017-2022 - Reproduction interdite

De (très) nombreuses fonctionnalités

- Gestion du versioning de code
- Suivi de bugs et collaboration
  - Milestones
  - Kanban
- Documentation interne
- CI/CD
  - Construction automatique
  - Gestion de pipelines (flux d'execution)
  - Gestion des livrables
  - Gestion des environnements de livraison
- etc.

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Historique du logiciel (1)

La vie de Gitlab

- 2011: **débuts** de GitLab
  - Créé par Dmitiy Zaporozhets, puis Veriy Sizov et Sytse Sijbrandij
  - Inspiré par Gitorious
- 2012: beta de *GitLab.com* et annonce sur HackerNews
- 2013: « I want to work on GitLab full time »
- 2014: GitLab Inc
- 2015: Incubation au Y Combinator
- 2016: Croissance
- 2020: The world's largest all-remote company
  - 1200 salariés dans 65 pays
- ...

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Historique du logiciel (2)

À souligner

- premiere licorne (en partie) Ukrainienne valorisée à plus de 1 milliard de $ en 2018
- Utilisé par IBM, Sony, NASA, Alibaba, Oracle, le CERN, Boing, SpaceX, etc.

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Les différentes distributions

GitLab CE

- Community Edition, la distribution libre (licence MIT)

GitLab EE

- Enterprise Edition, la distribution propriétaire (fonctionnalités supplémentaires)

Gitlab.com

- le service de forge en ligne, basé sur GitLab EE

Glenn ROLLAND © 2017-2022 - Reproduction interdite

GitHub vs GitLab

Similarités

- Outil « intégré »
- Navigation dans le code
- Gestion des tâches
- Gestion des livrables
- Sites statiques
- Intégration continue
  - Gitlab: depuis le début
  - Github: fonctionalité tardive
    - d'abord externe (ex: Travis-CI)
    - puis Github Actions

Différences

- Licence et business-model
  - OpenSource (Gitlab)
  - Propriétaire (Github)
- Auto-hébergement (Gitlab)
- Permissions plus fines (Gitlab)
- Terminologie (Github/Gitlab)
  - Pull Request / Merge Request
  - Gist / Snippet
  - Repository / Project
  - Organisation / Group
- Utilisation et interface utilisateur

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Utilisateurs, rôles et projets

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Premiers pas

- Création d'un compte utilisateur
- Création d'une clé SSH

```
ssh-keygen -t rsa -C moi@projet_rsa -f moi@projet_rsa
```

- Autorisation de la clef (publique) SSH sur Gitlab

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Tour d'horizon des projets

- Page par défaut
  - Configurable selon le projet
- Groupes / Projets
- Activité
- Création
- Forks

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Gestion des utilisateurs

- Gestion des équipes
- Invitation d’utilisateurs
- Les rôles
  - Définition des rôles
  - Droits associés aux roles

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Dépot de code (repository)

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Navigateur de fichiers

- Changement de branches
- Création de fichiers
- Création de branches
- Recherche de fichiers
- Téléchargement du code
- Clone

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Historique des commits

- Changement de branches
- Recherche par auteur
- Recherche par message
- Navigation à la version X

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Branches

- Niveau d’activité
- Comparaison
- Demande de fusion
- Téléchargement
- Création / Suppression

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Tags

- Listing
- Filtrage par nom
- Tri par date

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Contributeurs

- Timeline - Activité globale
- Timeline - Activité par contributeur
- Vue par branche

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Graphe

- Suivi des développements en parallele
- Suivi des fusions

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Comparaison de versions

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Des réglages spécifiques

- **Pour garantir un workflow**
  - Branche par défaut
  - Branches protégées
  - Tags protégés
- **Pour le déploiement**
  - Tokens de déploiement
  - Clefs de déploiement
- **Et aussi**
  - Mirroir automatique d’un dépot
  - Nettoyage automatique
- 📘 Dans **Settings > Repository**

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Tâches (issues)

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Gestion des taches

- Listing
  - Taches ouvertes
  - Taches fermées
  - Toutes les tâches
- Création
- Edition
- Import / Export
- Points d’étapes (milestones)
  - Listing
  - Créaztion
  - Filtres (par date, etc.)
- Labels et issue board
  - Project Information > Labels
- Commentaires spéciaux

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Autres fonctionalités

- Markdown
- Wiki
- Editeur web intégré
  - Staging et commit depuis l’IDE web
  - Navigation
- Releases

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Références

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Références (1)

Vidéos

- Josh Samuelson: Learning Gitlab
- [Josh Samuelson: Continuous](Integration and Continuous Delivery with GitLab https://www.linkedin.com/learning/continuous-integration-and-continuous-delivery-with-gitlab/learn-continuous-integration-and-delivery-with-gitlab?autoplay=true&resume=false&u=56203921)

Cours

- GitLAb: pour bien commencer https://www.fil.univ-lille1.fr/~routier/enseignement/licence/poo/tdtp/gitlab.pdf
- Groupe IPSL GitLab IN2P3 https://documentations.ipsl.fr/MESO_User/Documents/20200528-gitlab-ipsl.pdf

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Références (2)

Awesome Gitlab

- andorka/awesome-gitlab
- fkromer/awesome-gitlab

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Gitlab-CI

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Architecture de Gitlab-CI

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Définition et concepts (1)

Pipeline

- Un flux passant à travers une série d'étapes (*Stage*)
- Ils sont déclenchés par `git push` par défaut
- Ils peuvent aussi être déclenchés en externe (HTTP POST), utile pour le CI multi-projets
- Ils peuvent être programmés à l'avance et de façon régulière (*cron-like*)

Stage

- Une étape du *Pipeline*
- Les *Stages* sont évalués de façon séquentielle
- Un *Stage* exécute des *Jobs*

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Définition et concepts (2)

Job

- Le plus petit composant d'un *Pipeline*
- Les *Jobs* d'un même *Stage* sont exécutés en parallèle.
- Chaque *Job* contient une ou plusieurs commandes qui doivent être exécutées (séquentiellement)
- Il peut être paramétrée par des variables
- Il peut être limité ou contraint par règles (certains tags, certaines branches spécifiques, etc.)
- Il produit des fichiers « résultats »
- L’exécution des jobs est réalisée dans des *conteneurs* sur n’importe quel machine ou *Pod Kubernetes* disponible et enregistrés comme *GitLab Runner*

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Définition et concepts (3)

Artifacts

- Résultats d’une exécution gardé en mémoire pour traitement dans le *Pipeline*.

Cache

- Le résultats d’une exécution gardé temporairement en mémoire pour traitement dans le *Pipeline*

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Components de Gitlab-CI

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Composants (1)

Composants obligatoires

- GitLab Coordinator
- GitLab Runners
  - Runner Executors
- Composants tiers
  - Base de données (PostgreSQL)
  - Cache (Redis)
  - Gestion des dépôts Git (Gitaly)
  - Message Queue (Sidekiq)

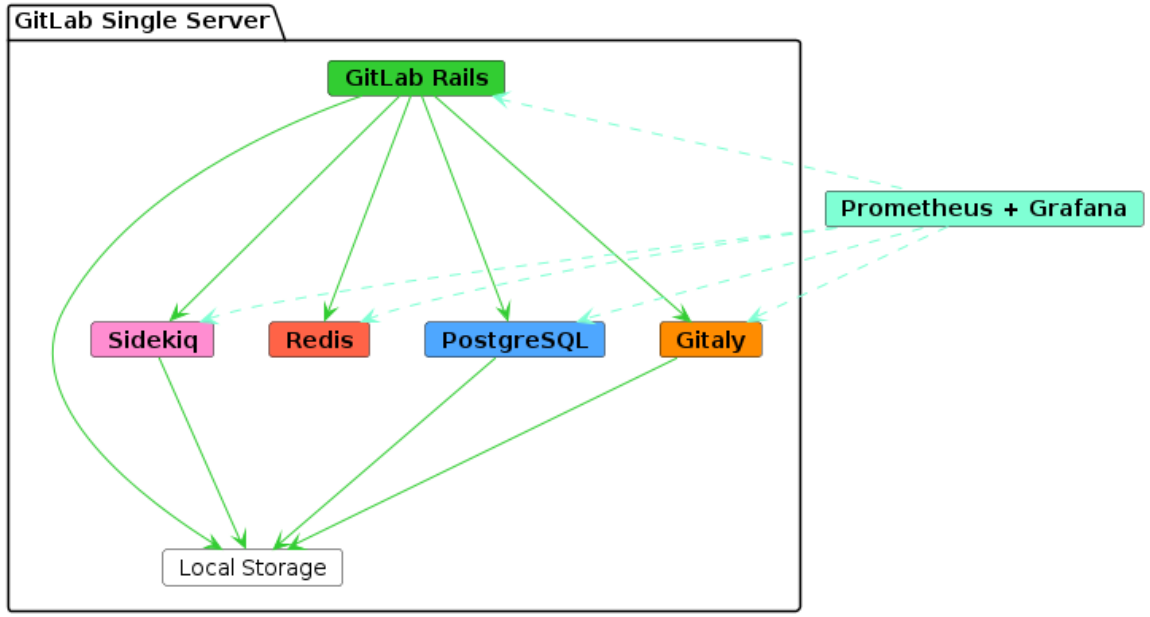
Composants optionnels

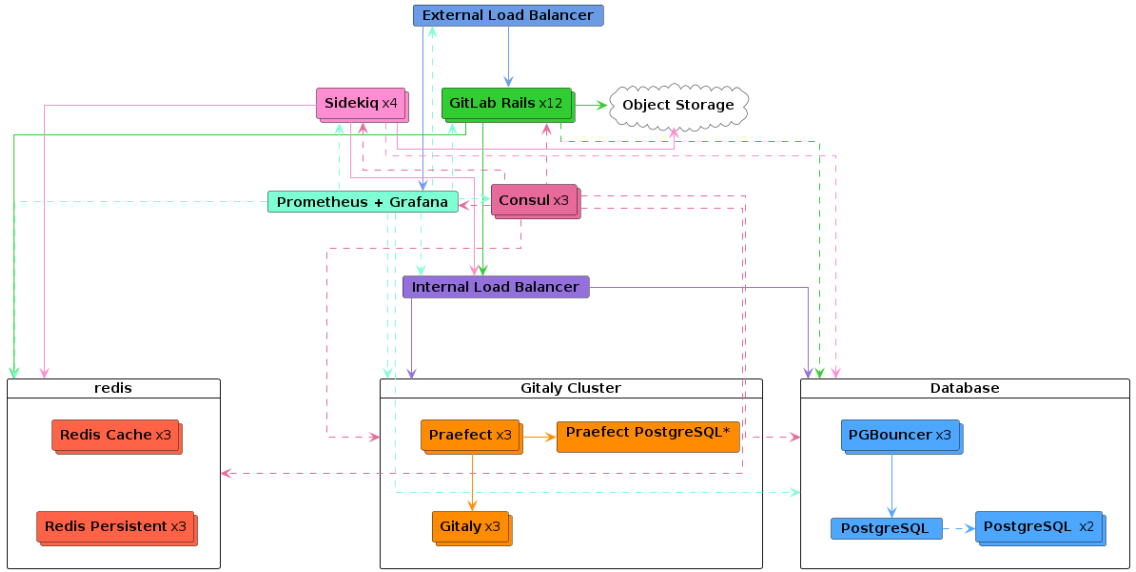
- Composants tiers
  - Configuration (Consul)
  - Stockage Objet (S3)
  - Load-Balancer Internes (HAProxy)
  - Reverse-proxy PostgreSQL (PGBouncer)

ℹ️ GitLab: Reference architecture

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Composants (2)





Composants (3)

GitLab Runner

- Agent léger et redimensionnable
- Récupère une tâche CI via l'API du coordinateur de GitLab CI/CD
- Exécute la tâche
- Renvoie le résultat à l'instance GitLab
- Il est très déconseillé de faire fonctionner les *Gitlab Runners* sur la même machine que le serveur

Runner Executor

- L'environnement dans lequel le travail sera exécuté par le *GitLab Runner*
- Les *Runner Executor* peuvent être: SSH, Shell, VirtualBox, Parallels, Docker, Kubernetes, Custom
- Chaque *GiLab Runner* définit au moins un exécuteur

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Installation

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Installation de Gitlab

Ressources recommandées

- CPU : 4
- RAM : 4 GB
- du stockage en conséquence

ℹ️ Ressources minimales recommandées selon la documentation de Gitlab (2022-07)

Étapes d'installation

- Gitlab: Install gitlab
- Gitlab: Update gitlab
- GitLab Docker images

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Intégration Continue

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Comment faire un pipeline

- 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`.

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Le fichier .gitlab-ci.yml

Syntaxe et mots-clés

- `script`
- `before_script` et `after_script`
- `image`
- `stages`
- `only` et `except`
- `only` avec `schedules`
- `when`
- `allow_failure`
- `tags`
- `services`
- `environment`
- `variables`
- `caches`
- `artifacts`
- `retry`

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Le fichier .gitlab-ci.yml

- Exemple avec gitbook
- Exemple avec Jekyll
- Exemple avec MkDocs
- Exemple avec Docker

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Contrôle des Jobs

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Control des Jobs

- Gitlab CI: Migrer depuis les only / exept vers les rules
- Gitlab CI: Lancer un job seulement sur des tags spécifiques
- Gitlab CI: Lancer une job seulement quand certains fichiers sont modifiés
- Job control

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Organiser un pipeline

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Livraison Continue

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Livraison Continue

- FIXME Registries
  - FIXME Packages
  - FIXME Containers
- Références

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Déploiement Continu

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Déploiement continu

- Gestion des Secrets
- Gestion des Environnements
- Automatiser les déploiements
- Livrer en production
- Feature flags
- Releases

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Gitlab Pages

- Permet de publier des sites web statiques
- Directement à partir d'un dépôt dans GitLab
- Un *Job* spécial nommé `pages` génère tous les `artifacts` d'un site web dans le dossier spécial `public/` .
- Example avec GitBook

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Avancé

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Éviter la duplication de code

1. On peut utiliser des ancres et des références YAML
2. On peut étendre sa propre image
3. Les mots-clés `before_script` permettent de factoriser une séquence de commandes
4. Utiliser des outils comme `make` ou `maven`

```yaml
image: ubuntu:latest

before_script:
- apt-get update
- apt-get install maven openjdk-8-jre

phase1:
  script:
    - cd tp1/
    - mvn test
phase2:
  script:
    - cd tp2/
    - mvn test
```

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Autres sujets intéressants

- Accélérer le pipeline
- Optimisation du cache
- Executer un job en local
- Mise en place d'un GitLab Runner
- Images Docker avec Gitlab-CI

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Références

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Architecture

- Gitlab Docs: Runners
- GitLab Docs: Executors
- Valentin Despa: A Brief Guide to GitLab-CI Runners and Executors

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Pipelines

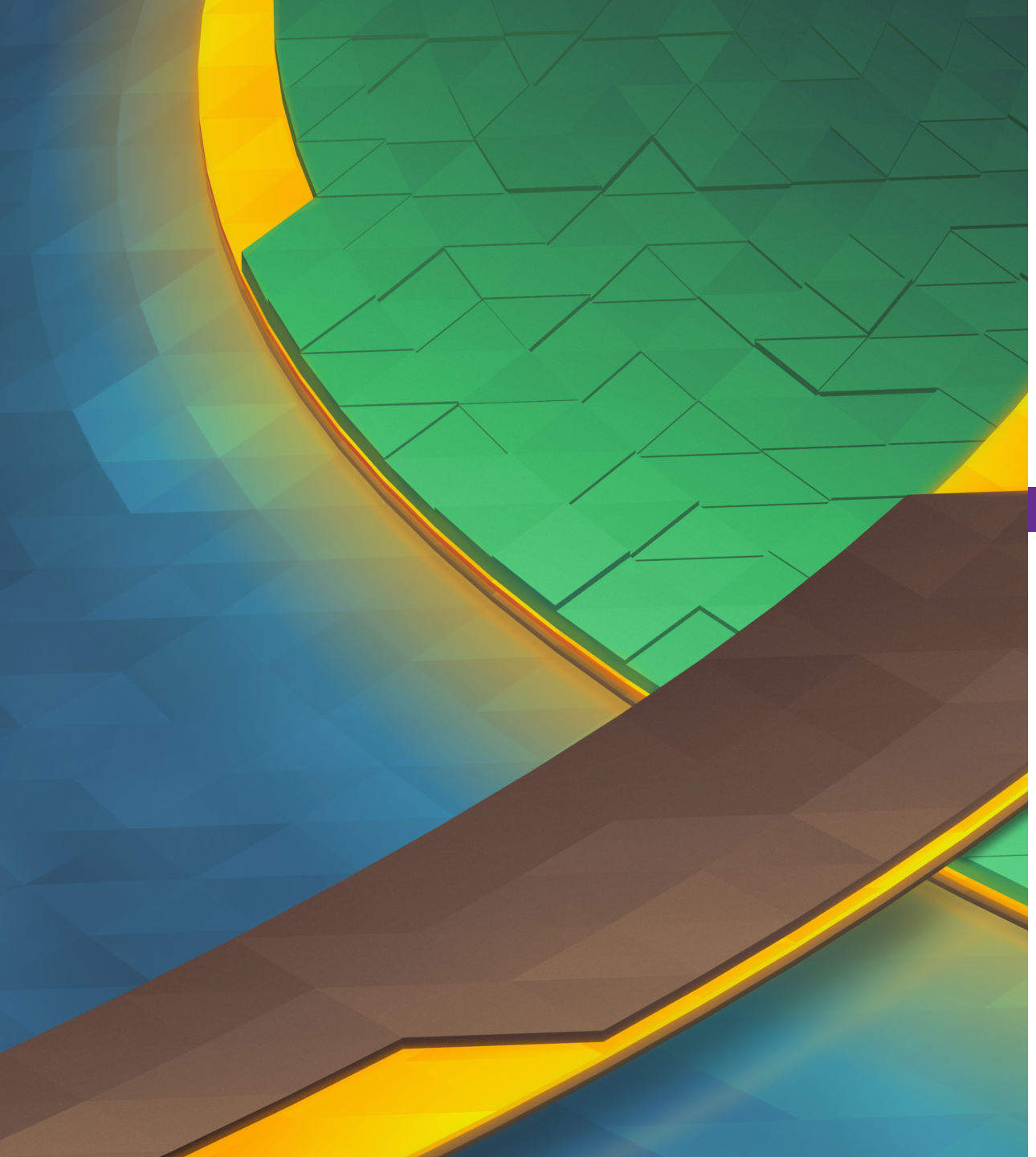
- Syntax of .gitlab-ci.yaml
- .gitlab-ci.yml keyword reference
- François-Emmanel GOFFINET: Intégration continue avec GitLab-CI
- Gitlab for Dummies

Glenn ROLLAND © 2017-2022 - Reproduction interdite

Livraison et déploiement

- Gitlab Docs: Releases

Glenn ROLLAND © 2017-2022 - Reproduction interdite



Merci pour votre attention !

Glenn ROLLAND © 2017-2022 - Reproduction interdite