

PLMlab

Philippe Depouilly, Henri Massias

27 Septembre 2017



Plan

Gitlab

PLMlab

PLMlab vs gitolite

Mattermost

Exemple d'utilisation

CI : Intégration continue

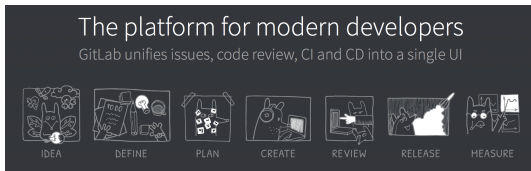
Exemples de génération de fichiers statiques

Utilisation du Registry pour stocker des containers Docker

Un exemple avancé

Introduction à Gitlab

Outil de gestion de projets basé sur git ("From idea to production")



- ▶ Possibilité de l'associer à Mattermost : équivalent de Slack, messagerie instantanée ++
- ▶ Voir la présentation de gitlab par les collègues de l'IRMA lors des journées Mathrice de Strasbourg.
- ▶ Y compris leur excellent Tutoriel vidéo de gitlab "Suivre et partager des documents LaTeX".

- ▶ Version de Gitlab installée : actuellement gitlab-ce 9.5.4, mise à jour en continu
- ▶ VM Ubuntu 16.04, 4 Go de RAM, 2 CPU
- ▶ installation Omnibus
- ▶ 31 utilisateurs, 52 projets
- ▶ Service en production, mais encore peu documenté
- ▶ Nécessite une première connexion à PLMlab avant de pouvoir ajouter un utilisateur à un projet
- ▶ Synchronisation des clefs ssh entre le compte PLM et le compte PLMlab (*fonctionnalité à venir*)
- ▶ Accès aux dépôts git via ssh (clefs ssh) ou https (login/mdp du compte PLM)



PLMlab vs gitolite

- ▶ PLMlab possède beaucoup plus de fonctionnalités que gitolite avec une interface ergonomique.
- ▶ PLMlab remplacera l'instance gitolite de la PLM à terme.
- ▶ PLMlab répond au projet de Forge logicielle au sein de la PLM.
- ▶ GitLFS permet de stocker de gros fichiers qui changent peu (images ISO, vidéos, ...)

Le futur de PLMlab

- ▶ Enrichir avec de nouvelles fonctionnalités
- ▶ Faire vivre des communautés : calcul, latex, developpeurs, ...
- ▶ Nouveaux usages auxquels on ne pense pas encore.



Mattermost en quelques mots

- ▶ Fonctionnalités de messagerie instantanée
- ▶ Possibilité de passer certaines commandes gitlab : créer une "issue", ...

The screenshot displays the Mattermost web interface. On the left is a sidebar with navigation options like 'CHANNELS' and 'PRIVATE GROUPS'. The main area shows a post from the 'Palo Alto' channel with a photo of people at a table and a code block for a shell script. The code block is titled 'Sample code for demo:' and contains a script to set and extract locale variables. On the right, a 'SEARCH RESULTS' panel shows search results for 'Community Heartbeat' and a specific issue titled '#20 Carriage returns not rendering'.

Post Content:

Fun times with OSLab, YC Canadians and Winter '15a. Sharing the Tension, midnight birthday party, trying to bypass facial recognition system to get snacks out of robotic cat treat dispenser

Code Block:

```
Sample code for demo:
before_action :set_locale
def set_locale
  $$.locale = extract_locale_from_tid || $$.default_locale
end
# get locale from top-level domain or return all if such locale
# You have to put something like:
# 127.0.0.1 application.com
# 127.0.0.1 application.fr
# 127.0.0.1 application.pl
# in your /etc/hosts file to try this out locally
```

Search Results:

October 15, 2015
Community Heartbeat
gitlab 11:09 PM June
Hi folks, adding v'all to the demo room just so you can see how things look, might need your help to take part and comment on some things to make screenshots...

October 15, 2015
Community Heartbeat
gitlab 10:05 PM June
#20 Carriage returns not rendering
GitLab.org / gitlab-mattermost: issue created by jasonm · Oct 11, 2015 - 12:54 PM
When markdown text has carriage returns, I expect the rendered text to have carriage returns, but it doesn't. I'm trying markdown with three carriage returns:



La gestion collaborative d'une conférence

Projet pour l'organisation d'une conférence (*Patrick Massot (CMLS, Palaiseau)*)

- ▶ gérer les sources du site web
- ▶ lister les problèmes d'organisation de la conférence en utilisant le issue tracker. En particulier créer une issue pour chaque demande de financement d'un participant.
- ▶ avoir un wiki qui sert de pense-bête
- ▶ Mise en garde sur l'édition des fichiers via l'interface web.

CI : Intégration continue

Définition Wikipédia

- ▶ **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.
- ▶ En pratique : L'intégration continue repose souvent sur la mise en place d'une brique logicielle permettant l'automatisation de tâches : compilation, tests unitaires et fonctionnels, validation produit, tests de performance

Dans PLMLab, ça s'appelle gitlab-ci.

- ▶ Intégré à gitlab
- ▶ Utilise des runners : shared ou specific runners
 - ▶ Créer un `.gitlab-ci.yml` à la racine du repo
 - ▶ en général 3 étapes : build, test, et deploy
 - ▶ Exemples de configuration
 - ▶ Quickstart de gitlab-ci
- ▶ GitLab et les runners communiquent via une API : le seul prérequis est que la machine sur laquelle est installé le runner ait accès à Internet

Démo de CI avec un projet LaTeX

Démonstration de la création d'un projet avec un document LaTeX et compilation automatique

Génération de pages web statiques

- ▶ Fonctionnalité "Pages" de gitlab
- ▶ Met à disposition uniquement des pages web statiques : HTML, CSS et Javascript
- ▶ On peut utiliser un générateur de site web statique : Jekyll, Middleman, Hexo, Hugo, Pelican
- ▶ Créé un site web pour le projet accessible sur `http://nom_du_groupe.pages.math.cnrs.fr/nom_du_projet` ou `http://username.pages.math.cnrs.fr/nom_du_projet`
- ▶ Possibilité d'utiliser https
- ▶ Les "Pages" de gitlab sont publiques
- ▶ Pour l'utiliser dans un projet où il y a aussi du code, faire une branche pages dans le repo (puis indiquer only: - pages dans `.gitlab-ci.yml`)



Démonstration

Fonctionnalité Wiki de gitlab

- ▶ Basé sur gollum
- ▶ Affecte les mêmes droits qu'au dépôt principal
- ▶ Dépôt séparé

```
GitLab / pres-Mathrice / plmlab ▾  
Wiki  
  
Clone repository pres-Mathrice/plmlab.wiki  
SSH git@plmlab.math.cnrs.fr:pres-Mathrice/plmlab.wiki  
  
Install Gollum  
gem install gollum  
It is recommended to install github-markdown so that GFM features render locally:  
gem install github-markdown  
  
Clone your wiki  
git clone git@plmlab.math.cnrs.fr:pres-Mathrice/plmlab.wiki.git  
cd plmlab.wiki  
  
Start Gollum and edit locally  
gollum  
== Sinatra/1.3.5 has taken the stage on 4567 for development with backup from Thin  
>> Thin web server (v1.5.0 codename Knife)  
>> Maximum connections set to 1024  
>> Listening on 0.0.0.0:4567, CTRL+C to stop
```

Pages + Wiki vont remplacer très prochainement la ferme de Wiki :

<https://wiki.math.cnrs.fr>



Utilisation du Registry pour stocker des containers Docker

▶ Permet de conserver des containers Docker

[GITLab](#) / [docker-images](#) / [ubuntu](#) ▾
Container Registry

Container Registry
 With the Docker Container Registry integrated into GITLab, every project can have its own space to store its Docker images.
[Learn more about Container Registry.](#)

How to use the Container Registry

First log in to GITLab's Container Registry using your GITLab username and password. If you have 2FA enabled you need to use a [personal access token](#).











```
docker login plmlab.math.cnrs.fr:4567
```

Once you log in, you're free to create and upload a container image using the common **build** and **push** commands:

```
docker build -t plmlab.math.cnrs.fr:4567/docker-images/ubuntu .
docker push plmlab.math.cnrs.fr:4567/docker-images/ubuntu
```

Use different image names
 GITLab supports up to 3 levels of image names. The following examples of images are valid for your project:

```
plmlab.math.cnrs.fr:4567/docker-images/ubuntu:tag
plmlab.math.cnrs.fr:4567/docker-images/ubuntu/optional-image-name:tag
plmlab.math.cnrs.fr:4567/docker-images/ubuntu/optional-name/optional-image-name:tag
```

- [docker-images/ubuntu](#)  
- [docker-images/ubuntu/ubuntu/ubuntu](#)  
- [docker-images/ubuntu/centos/centos](#)  
- [docker-images/ubuntu/ubuntu](#)  
- [docker-images/ubuntu/centos](#)  

Génération et déploiement de containers à partir d'un projet PLMLab

- ▶ Création de containers Docker à l'aide d'un runner shell
- ▶ Création de containers Docker à l'aide de dind (Docker in Docker)
- ▶ On automatise la création de containers Docker
- ▶ Comme pour Pages, pour l'utiliser dans un projet où il y a aussi du code, faire une branche spécifique dans le repo

Pour finir

Des questions? Des besoins?



Utilisation de gitlab-runner pour configurer son propre runner

```
$ gitlab-ci-multi-runner install  
$ gitlab-ci-multi-runner start  
$ gitlab-ci-multi-runner list  
Listing configured runners      ConfigFile=/Users/hmassias/.gitlab-
```

Utilisation de gitlab-runner pour configurer son propre runner

```
$ gitlab-ci-multi-runner register
Please enter the gitlab-ci coordinator URL (e.g. https://gitlab.com):
  https://plmlab.math.cnrs.fr/
Please enter the gitlab-ci token for this runner:
  xxxxxxxxxxxxxxxxxxxxxxxxxxxx
Please enter the gitlab-ci description for this runner:
  Aldur
Please enter the gitlab-ci tags for this runner (comma separated):
  latex
Whether to run untagged builds [true/false]:
  [false]: true
Whether to lock Runner to current project [true/false]:
  [false]: true
Registering runner... succeeded runner=m5xA8NcR
Please enter the executor: shell, ssh, virtualbox, docker+machine,
shell
Runner registered successfully. Feel free to start it, but if it's
```



Utilisation de gitlab-runner pour configurer son propre runner

```
$ gitlab-ci-multi-runner list  
Listing configured runners      ConfigFile=/Users/hmassias/.gitlab-  
Aldur      Executor=shell Token=xxxxxxxxxxxxxxxxxxx URL=https://plmla
```