

# GITLAB CI/CD

Thomas Aldaitz

*taldaitz@dawan.fr*



---

# INTRODUCTION

---

- Connaissances des commandes de base de GIT
- Droits d'administration sur sa machine

- Maîtriser Gitlab pour la gestion de ses **Repository** git
- Gérer ses projets de manière agile avec les **Issues** et **Merge Requests**
- Automatiser l'exécution de compilation et de tests à l'aide **Pipeline** complexe
- Déployer son application sur d'autres environnements à l'aide de **Runner**

---

# GIT / GITLAB

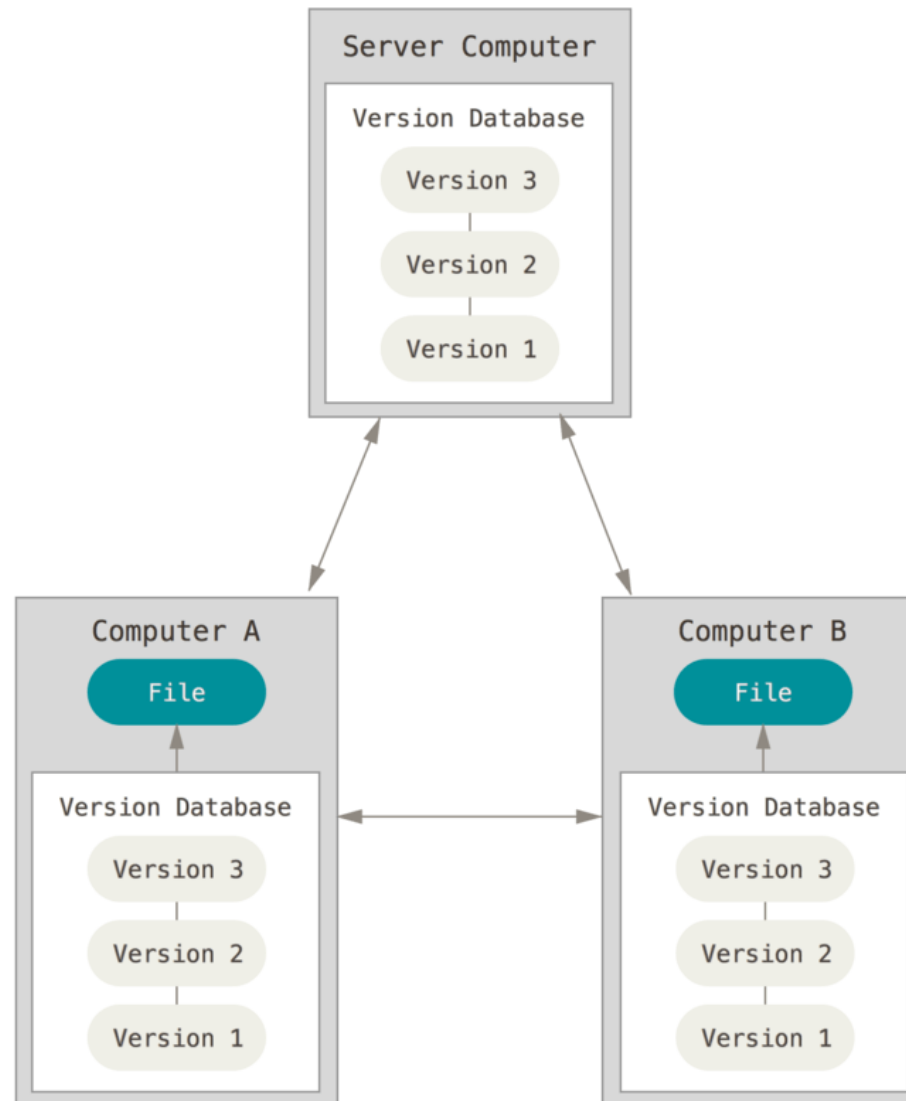
---

## GIT : RAPPELS

- Versionning
- Décentraliser
- Branche/feature



# ARCHITECTURE DISTRIBUÉ



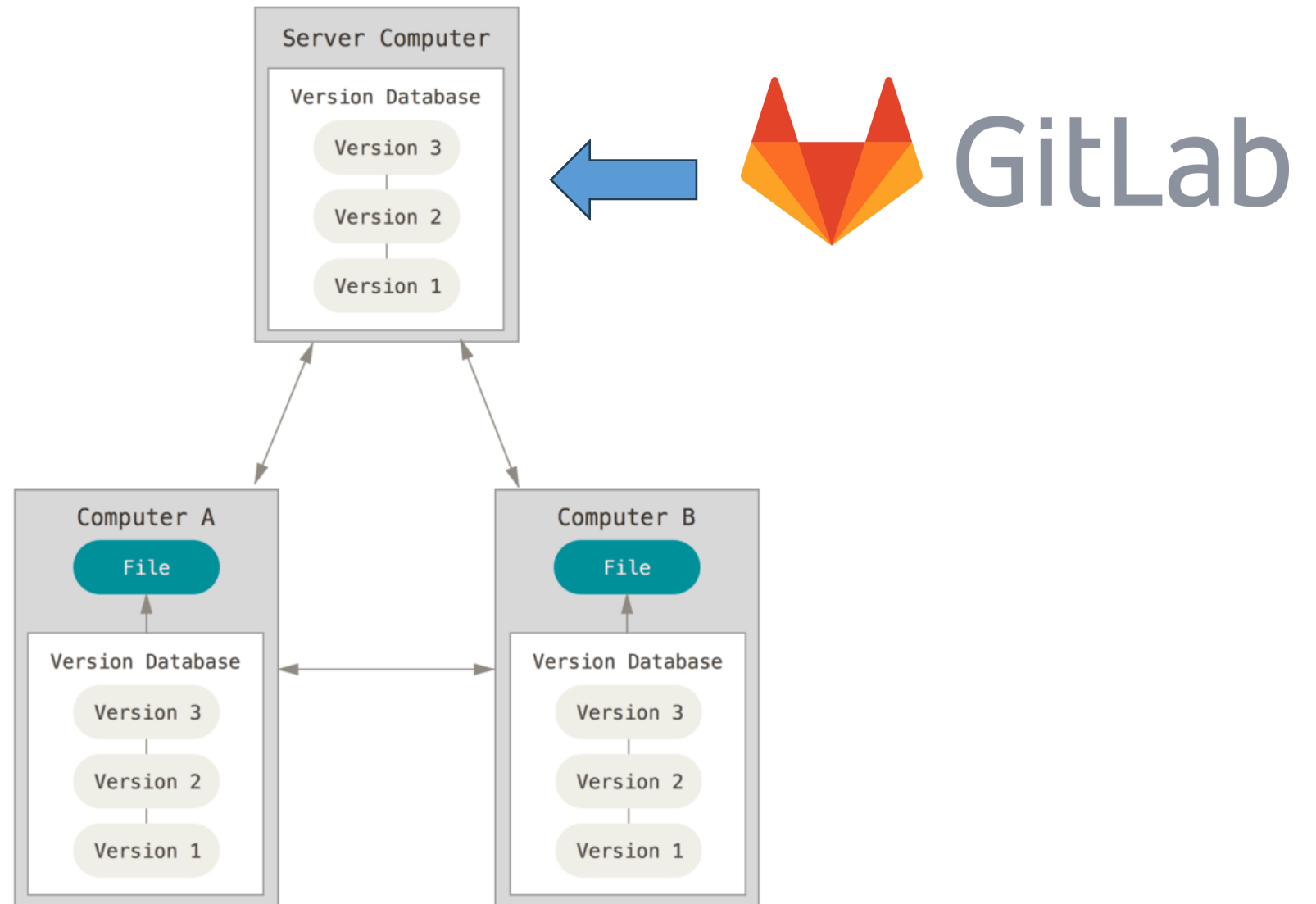
# GITLAB





- Plateforme
- Tickets/MR
- CI/CD






GitLab









Rechercher ou aller à...

Votre travail

Projets

Groupes

Tickets

requêtes de fusion

Liste des pense-bêtes

Jalons

Extraits de code

Activité

Espaces de travail

Environnements

Opérations

Sécurité

[Votre travail](#) > Projets

## Projets

Explorer les projets

Nouveau projet

Les vôtres 1Favoris 0Suppression en attente

Filtrer par nomLangueNom

TousPersonnels

T

dawan-aldaitz / training

Owner

✓

☆ 0

👤 0

🔗 1

📄 0

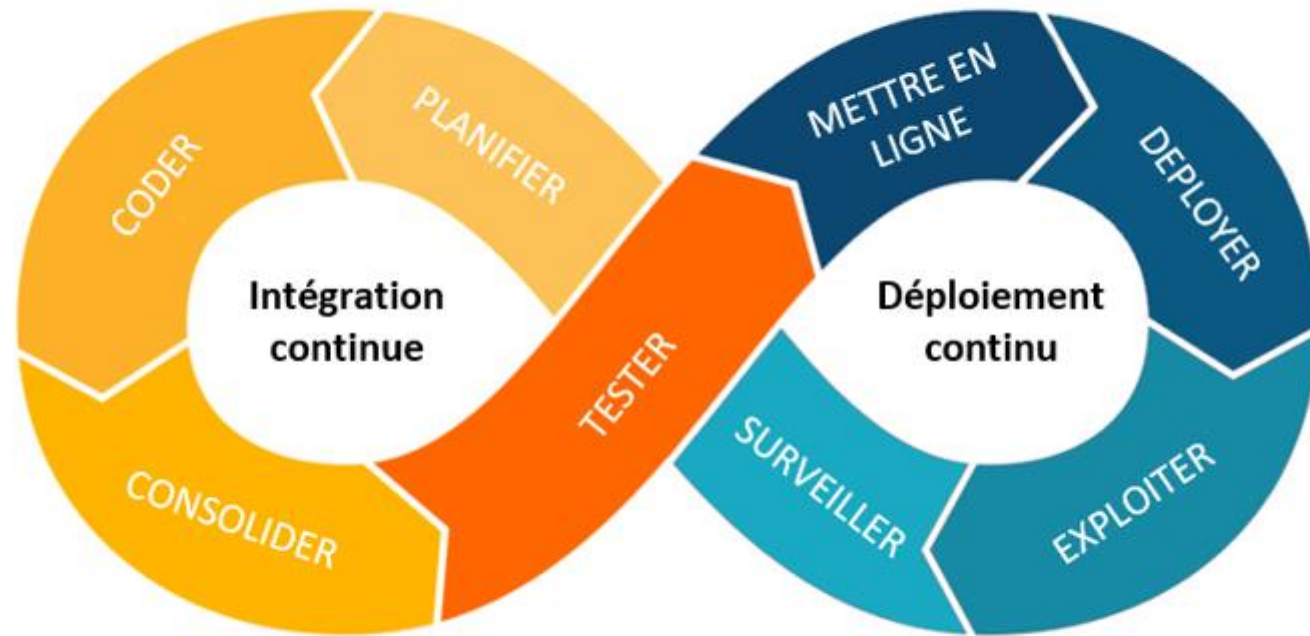
Mis à jour il y a 20 heures

---

# INTÉGRATION CONTINUE

---

# CI / CD



# L'eXtrem Programming

- Méthode de programmation agile présenté par **Kent Beck** en 1999.
- Ensemble de règles et de bonnes pratiques :
  - **Intégration continue**
  - **Factorisation**
  - Rythme soutenable
  - Programmation en binôme (*pilote/co-pilote*)
  - **Tests unitaires**
  - ...

# Objectifs de l'XP

- Améliorer la qualité du code, et donc de la solution développée
- Améliorer les conditions de travail des développeurs
- Permet des mises en production plus fréquentes sans mettre en péril l'existant
- Augmenter la synergie entre les équipes dev et les métiers

# Agilité

- Mouvement né en 2001 aux USA.
- Réunion de 17 experts en développement logiciel.
- But : trouver un socle commun de valeurs et de bonnes pratiques.
- Résultats : écriture du **Manifeste pour le développement logiciel Agile** :  
<http://agilemanifesto.org/iso/fr/manifesto.html>
- et création de l'**Agile Alliance** (association chargée de la promotion de l'agilité et du soutien aux équipes) :  
<http://www.agilealliance.org/>

---

# LA SOLUTION

---

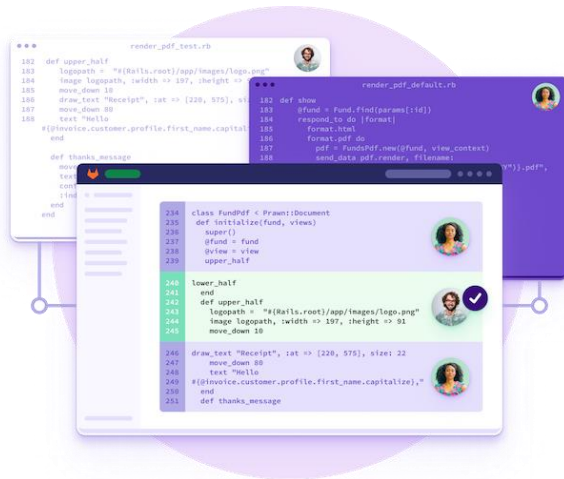


# HISTORIQUE

- Créée en 2011
- Projet Open-Source (+ 3000 contributeurs)
- + 30 millions d'utilisateurs (estimation)
- Aujourd'hui en **17.1**



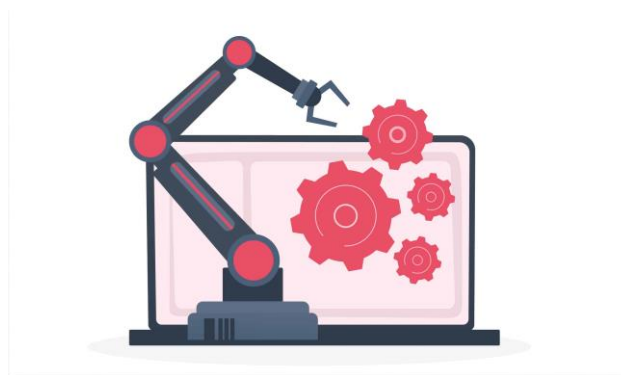
# LES FONCTIONNALITÉS



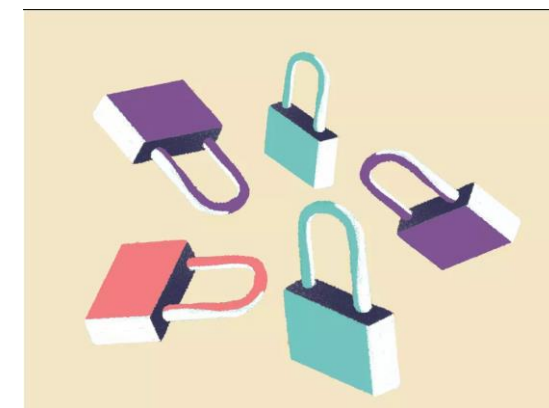
Partage du code



Gestion de projet



Automatisation



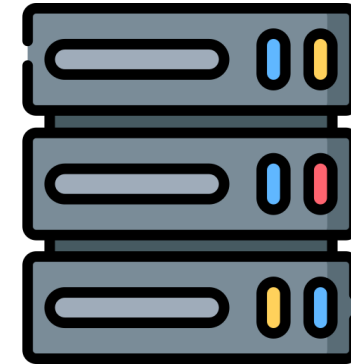
Sécurité

## Saas


























































- Hébergé hors SI
- Pas administration
- Mise à jour automatique





## Self Managed






- Hébergé localement
- Administration complète

# LE COUTEAU SUISSSE


 Manage	 Plan	 Create	 Verify	 Package	 Release	 Configure	 Monitor	 Secure
								
								
								
								
								
								


   


  


Q Rechercher ou aller à...


**Votre travail**


 Projets


 Groupes


 Tickets


 requêtes de fusion >


 Liste des pense-bêtes


 Jalons


 Extraits de code

 Activité

 Espaces de travail

 Environnements

 Opérations

 Sécurité >

Votre travail > Projets

## Projets

Explorer les projets

Nouveau projet










Les vôtres 2 Favoris 0 Suppression en attente

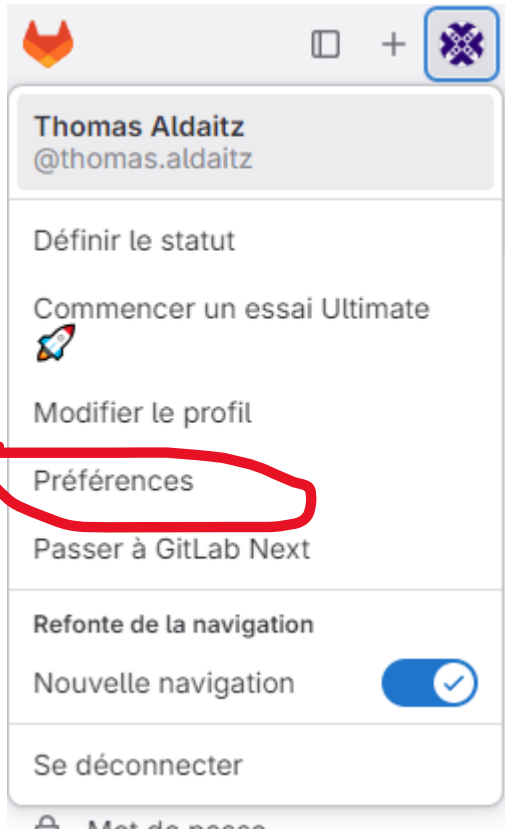
Filtrer par nom

Langue

Nom

Tous Personnels

I	dawan-aldaitz / irep  Owner	☆ 0  0  0  1	Mis à jour il y a une heure
T	dawan-aldaitz / training  Owner	 ☆ 0  0  1  0	Mis à jour il y a 22 heures



## Paramétrage du compte:

- Thème
- Coloration Syntaxique
- Diffs
- Langue de l'interface
- Premier jour de la semaine

# ATELIER

*Installation :*

<https://about.gitlab.com/install/#ubuntu>

*Vagrant :*

*vagrant init generic/ubuntu2204*

---

# CRÉATION D'UN REPO

---





## Créer un projet vide

Créez un projet vide pour, entre autres, stocker vos fichiers, planifier votre travail et collaborer sur le code.

- Projet Vierge -> Simple *git init*
- Possibilité d'ajouter un README




## Créer à partir d'un modèle

Créer un projet pré-rempli avec les fichiers nécessaires pour vous permettre de démarrer rapidement.

- Repo constitué d'une base de départ
- README détaillé
- Premiers éléments de CI/CD

# INTERFACE PROJET



1

1

1

Rechercher ou aller à...

Projet

I Irep

Apprendre GitLab 42%

Épinglé

Tickets 1

requêtes de fusion 0

Gestion

Programmation

Code

Compilation

Sécurité

Déploiement

Opération

Surveillance

Analyse

Paramètres

dawan-aldaitz > Irep

I Irep

Identifiant de projet : 50789779

🔔

☆ Ajouter aux favoris 0

🍴 Bifurcations 0

🔗 362 validations

🌐 1 branche

🏷️ 0 étiquette

💾 101,1 Mo stockage de projet

Merge pull request #66 from WildCodeSchool/fixcataloguepdf

aldaitzwild rédigé il y a 2 mois

Non vérifiée

fb2f63d6

dev irep /

Historique

Rechercher un fichier

Modifier

📄

Cloner

LISEZMOI

Ajouter une LICENCE

Ajouter un CHANGELOG

Ajouter un CONTRIBUTING

Activer Auto DevOps

Ajouter une grappe de serveurs Kubernetes

Configuration CI/CD

Ajouter une page wiki

Configurer les intégrations

Nom	Dernière validation	Dernière mise à jour
.github/workflows	Update main.yml	il y a 2 mois
assets	Fix route for catalogue / Center image on picture modal	il y a 2 mois
bin	Initial commit	il y a 3 mois
config	Variable for path to generate pdf	il y a 2 mois
migrations	stat page with good route for gallery and show directly ...	il y a 2 mois
public	redirect to admin page after crop	il y a 2 mois
src	resolves conflict	il y a 2 mois

---

# RÉCUPÉRATION DE REPO EXISTANT

---

# ATELIER

- *Récupérer Simple calculator :*  
<https://github.com/taldaitz/simpleCalculator>

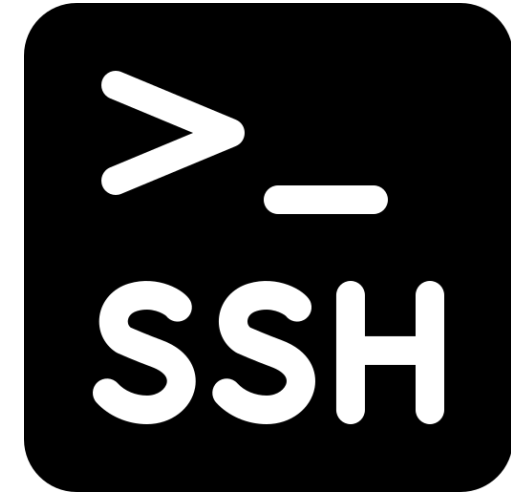
---

# CLÉS SSH

---



- Authentification Classique
- Idéale pour multipostes
- Peu sécurisé



- Authentification par Clés (publique/privée)
- Idéale pour poste nomade
- Plus sécurisé !!

Création de la clé :

```
ssh-keygen -t ed25519 -C "taldaitz@dawan.fr"
```

Récupération de la clé :

```
cat "C:\Users\<you_username>\.ssh\id_ed25519.pub"
```

# ATELIER

- *Créer sa clé SSH*  
(<https://docs.gitlab.com/ee/user/ssh.html>)
  - *L'associer à son compte Gitlab*
  - *Puis modifier notre page HTML*
    - *Et l'envoyer sur GitLab*





**La clé SSH a-t-elle  
bien été utilisé ?**



Le protocole utilisé dépend de la déclaration du repo distant !

```
C:\dev\formations\gitlab\firstproject (main -> origin)
λ git remote -v
origin https://gitlab.com/dawan-aldaitz/firstproject.git (fetch)
origin https://gitlab.com/dawan-aldaitz/firstproject.git (push)
```

```
C:\dev\formations\gitlab\0322_P3_Php_Lyon_Irep (dev -> origin)
λ git remote -v
origin git@gitlab.com:dawan-aldaitz/irep.git (fetch)
origin git@gitlab.com:dawan-aldaitz/irep.git (push)
```

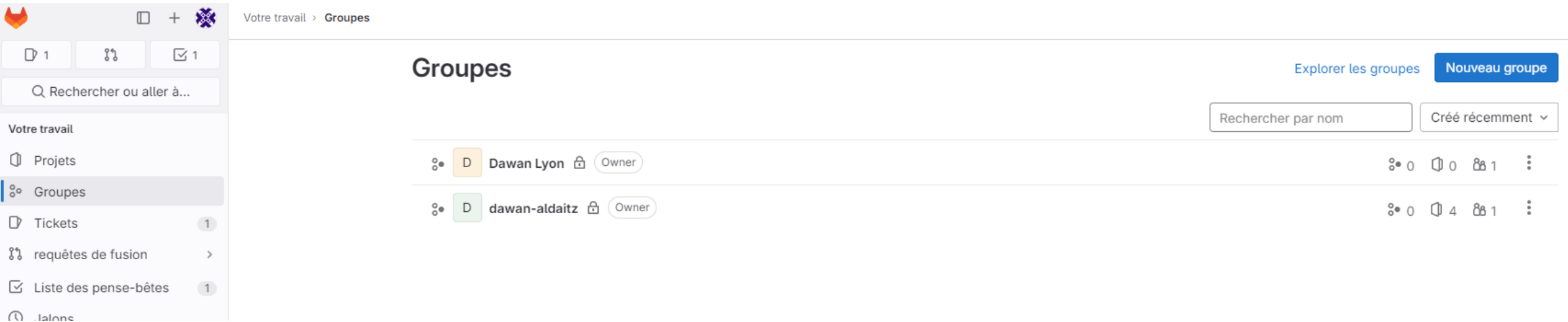
# ATELIER

- 1 - Créer en local une branche "dev"
- 2 - Ajouter un fichier HTML => error.html  
(Oup! quelque chose cloche)
  - 3 - Commit la modif
  - 4 - La pousser sur git lab
- 5 - Vérifiez que tout est bien remonté

---

# LES GROUPES

---

















Votre travail > Groupes

## Groupes

Explorer les groupes Nouveau groupe

Rechercher par nom Créé récemment

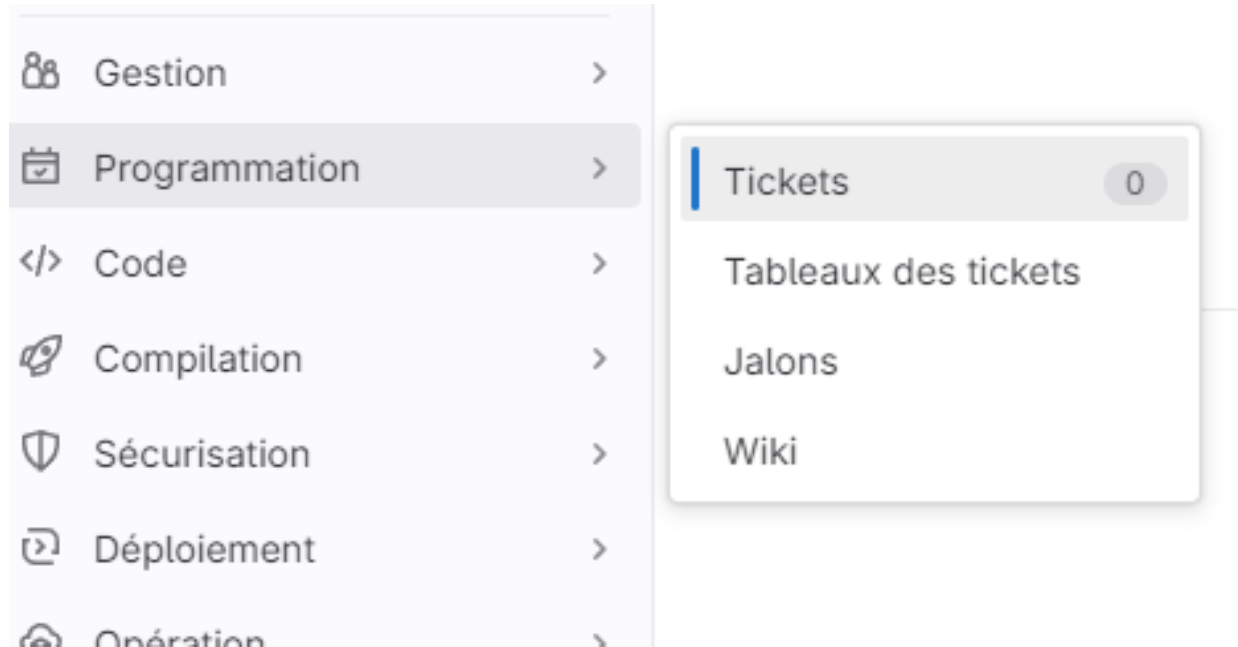
	 D	Dawan Lyon		Owner	 0	 0	 1	
	 D	dawan-aldaitz		Owner	 0	 4	 1	

- Offre la possibilité d'avoir un niveau d'organisation au dessus de l'organisation individuel
- Convient mieux aux équipe ou aux entreprise
- Limité en version gratuite à 5 personnes

---

# LES TICKETS

---



Les **tickets** sont la pierre angulaire de la gestion de projet sous Gitlab.

Tickets = User Stories

Gérés par un systèmes de labels.

À venir **Jalon** oct. 9, 2023–oct. 20, 2023

Fermer le jalon



## Sprint 1

ID du jalon : 4298125 

**Tickets** 1 requêtes de fusion 0 Participants 1 Labels 1

Tickets non commencés (ouverts et non attribués)

 0

Tickets en cours (ouverts et attribués)

 1

Tickets terminés (fermés)

 0





Créer une page d'accueil

#1 **En cours** 


Permet une gestion à l'échelle de l'itération.



# KANBAN BOARD



1



2

Rechercher ou aller à...

dawan-aldaitz > MyCalculator > Tableaux des tickets

Development

Rechercher

Afficher les labels

Modifier le tableau

Créer une liste

Projet

M MyCalculator

Apprendre GitLab 42%

Épinglé

Tickets 1

requêtes de fusion 0

Gestion

Programmation

Tickets 1

Tableaux des tickets

Jalons

Wiki

Code

Compilation

Ouvert 0

En cours 1

En Revue 0

Closed 0

Créer une page d'accueil

#1 oct. 21

On peut créer des modèles de rédaction pour les Tickets (et les Merge Request).

On peut également y inclure des raccourcis pour pré-configurer les tickets comme :

- /assign @user
- /cc @user
- /estimate\_time
- /due
- ...

([https://docs.gitlab.com/ee/user/project/quick\\_actions.html](https://docs.gitlab.com/ee/user/project/quick_actions.html))

```
main ▾ myCalculatorModel / .gitlab / issue_templates / issue_01.md

⋮ ▾ M+ issue_01.md 840 o
1  ## Summary
2
3  (Summarize the bug encountered concisely)
4
5  ## Steps to reproduce
6
7  (How one can reproduce the issue - this is very important)
8
9  ## Example Project
10
11 (If possible, create an example project here on GitLab.com that exhibits the proble
12 behavior, and link to it here in the bug report.
13 If you are using an older version of GitLab, this will also determine whether the b
14 in a more recent version)
15
16 ## What is the current bug behavior?
17
```

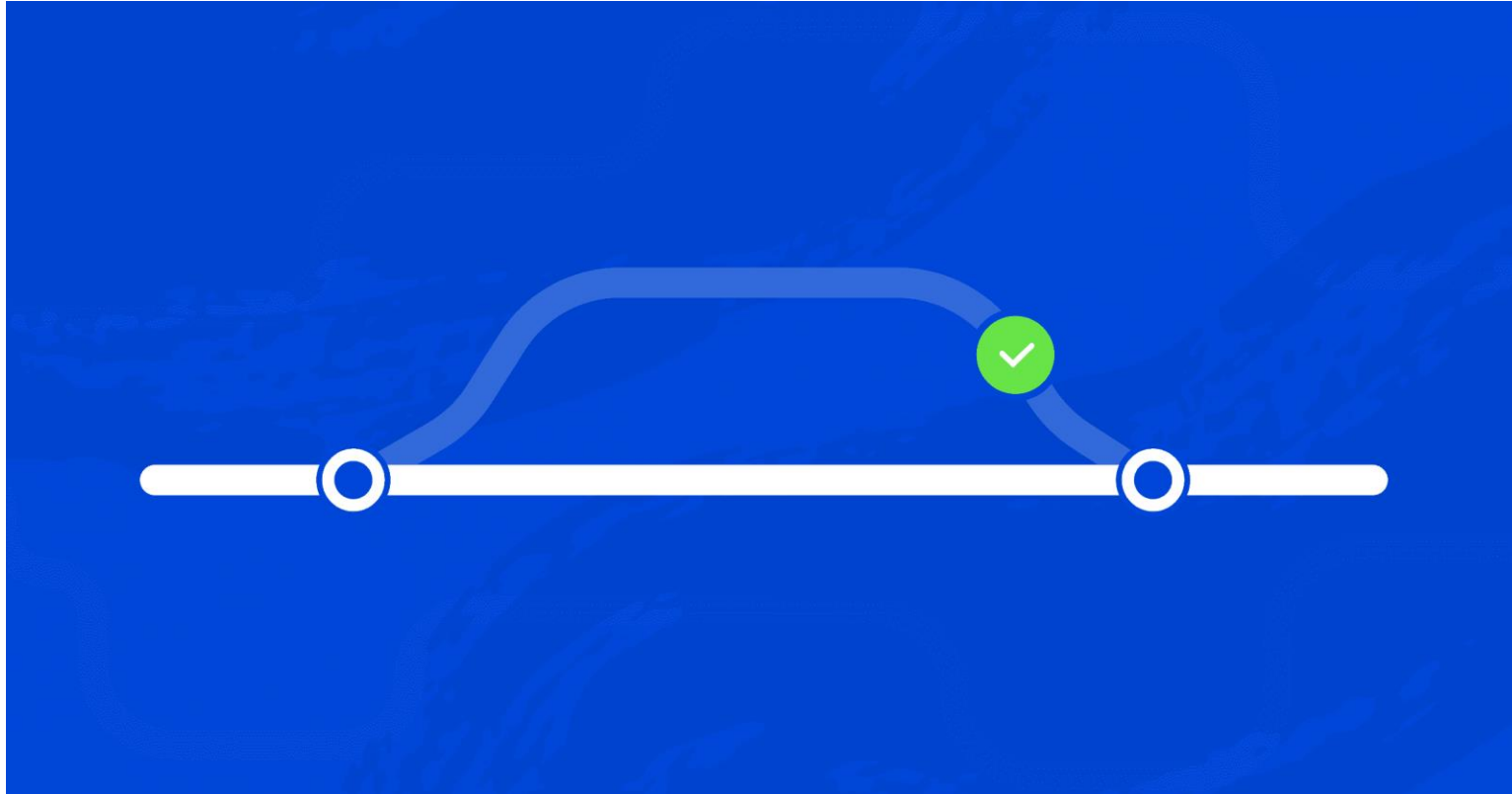
# ATELIER

- *Créer 3 labels : En cours, En revue, Abandonné*
  - *Créer 1 jalon*
  - *Créer 2 tickets*

---

# LES MERGE REQUEST

---



- Principe de vérification par les paires
- Pair programming = Qualité de code

Créer ses branches à partir des tickets pour s'intégrer dans le workflow

## Créer une page d'accueil

Modifier



 Ouvert  Ticket créé il y a 57 minutes par **Thomas Aldaitz**

Créer une page avec un carrousel, un menu responsive et deux paragraphes de présentation



Créer une requête de fusion



 Faites glisser vos designs ici ou [cliquez pour les téléverser](#).

# MERGE REQUESTS


dawan-aldaitz > MyCalculatorModel > requêtes de fusion > Nouveau

## Nouvelle requête de fusion

### Branche source

dawan-aldaitz/myCalculatorModel ▼

Sélectionner une branche sour... ▼

 Sélectionner une branche à comparer

Comparer les branches et continuer

### Branche cible

dawan-aldaitz/myCalculatorModel ▼

main ▼



Mettre à jour le fichier `.gitlab-ci.yml`

Thomas Aldaitz a rédigé oct. 03, 2023



3e6fec7d



# ATELIER

- *Depuis un ticket créer une branche*
  - *La rapatrier en local*
- *Réaliser une modification puis la pousser*
  - *Checker la MR puis la valider*



---

# ANALYSE

---

Next

2 2

Rechercher ou aller à...

Projet

- MyCalculatorModel
- Apprendre GitLab 42%
- Épinglé
- Tickets 2
- requêtes de fusion 0

Gestion

Programmation

Code

Compilation

Sécurisation

Déploiement

Opération

Surveillance

Analyse

Données d'analyse des...

Statistiques sur les con...

Données d'analyse du ...

Expériences du modèle

dawan-aldaitz > MyCalculatorModel > Données d'analyse des chaînes de valeur

## Données d'analyse des chaînes de valeur

⌚ Filtre les résultats

Du 2023-09-04 Au 2023-10-03 30 jours sélectionnés ⓘ

**Ticket <1m** Forfait - Code <1m Test - Examen 6m Étape de mise en place -

### Métriques du cycle de vie

Nouveaux tickets 2 Validations 28 Déploiement -

Tickets 2 éléments	Dernier événement	Durée
Nouveau ticket #2 · Créé about 1 hour ago par Thomas Aldaitz	about 1 hour ago	0 seconde
Créer une page d'accueil #1 · Créé about 1 hour ago par Thomas Aldaitz	about 1 hour ago	0 seconde

---

# CI / CD

---

Diagramme d'activités

---

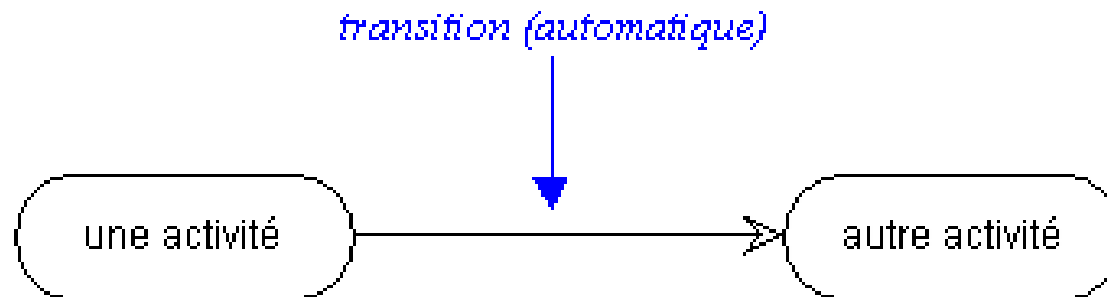
# SCHEMATISATION

---

Diagramme d'activités

# Description

- Représentation du déroulement d'un cas d'utilisation ou d'une méthode.
- Variante du diagramme d'états-transitions :
  - \* Les transitions sont automatiques entre la fin d'une étape et le début de la suivante.
  - \* Les diagrammes d'états-transitions mettent l'accent sur la traversée d'un processus (*process*) par un objet, alors que, les diagrammes d'activités se focalisent sur le flux d'activités concourant à la réalisation d'un processus

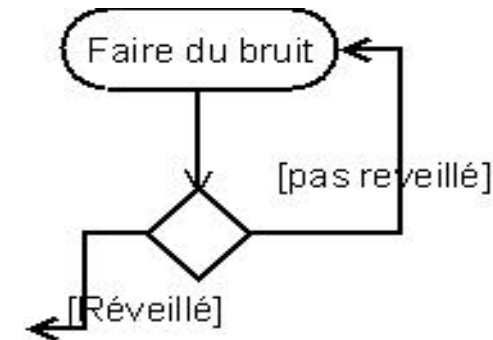


# Éléments du diagramme

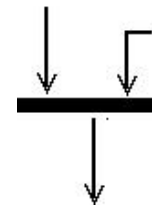
- Début et fin de flux d'activité



- Les transitions conditionnelles

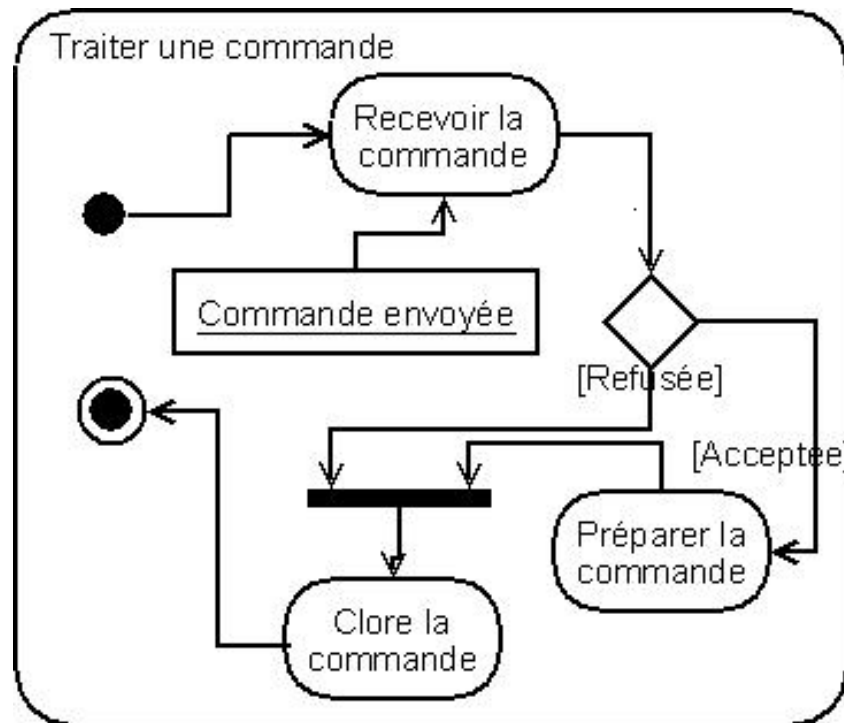


- Séparations et jointures

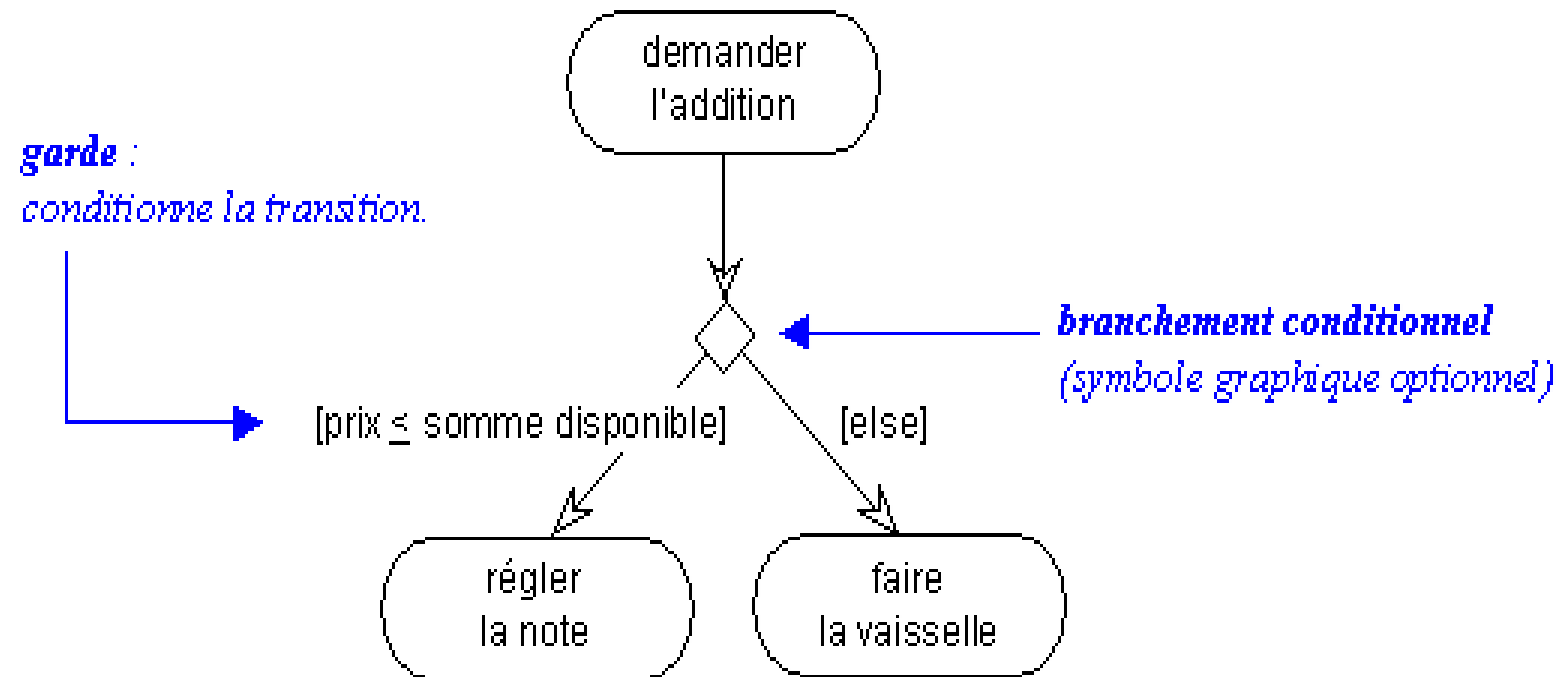


- Activité (séparable en parties), objets : rectangle

# Exemple

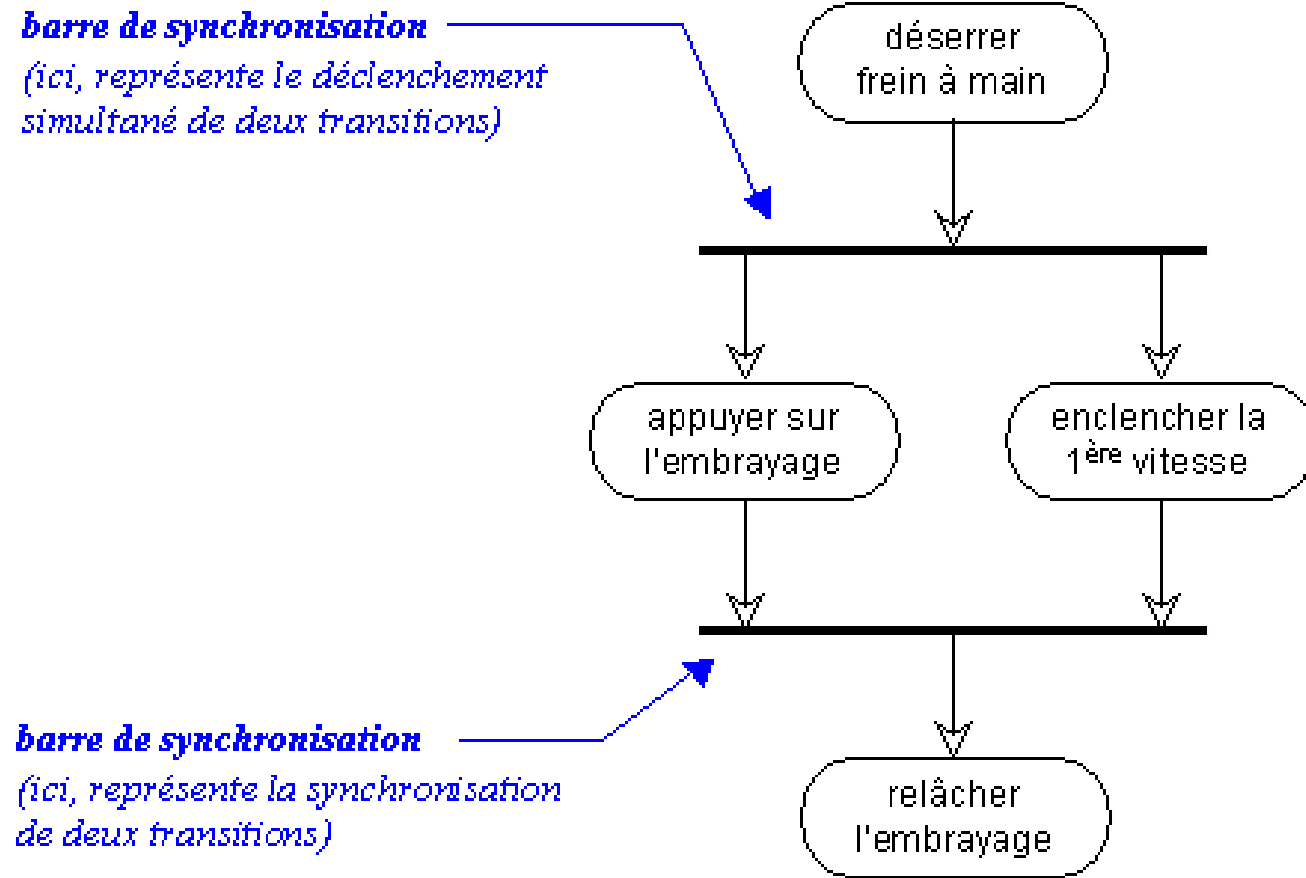


# Exemple (2)

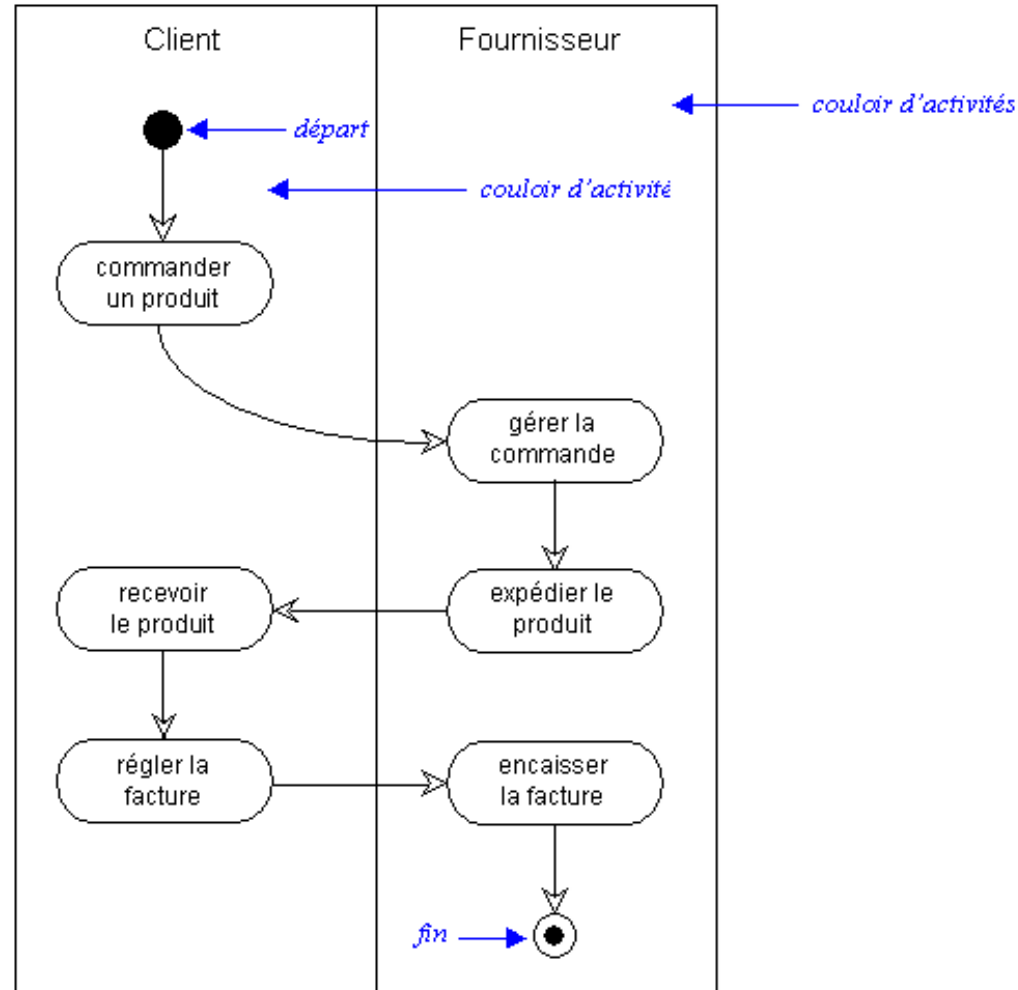




# Synchronisation et activités parallèles



# Couloirs d'activités



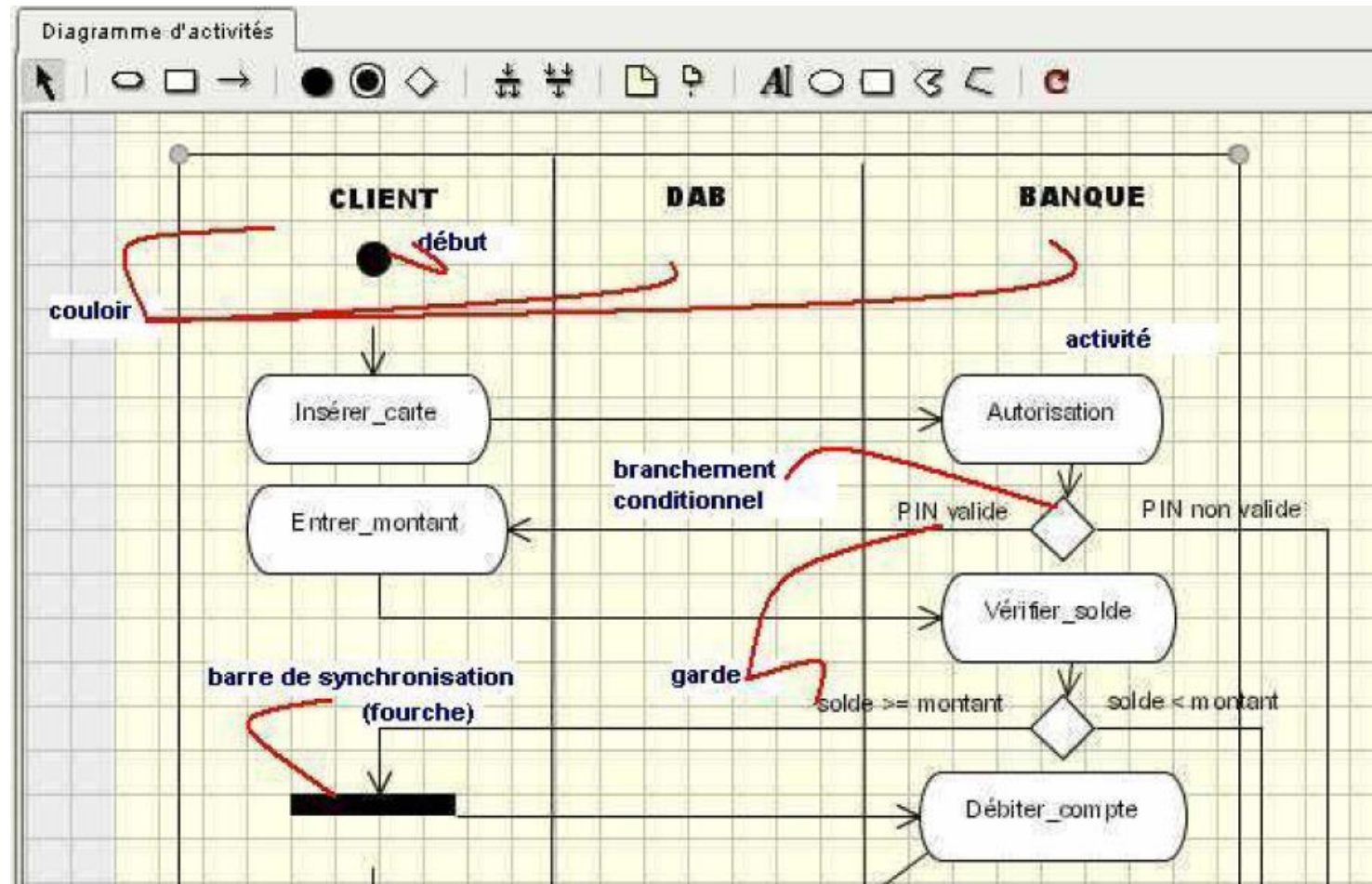
# Exemple de synthèse

■ On désire créer un diagramme d'activités pour l'opération de retrait d'argent à partir d'un distributeur de billets. 3 entités interviennent :

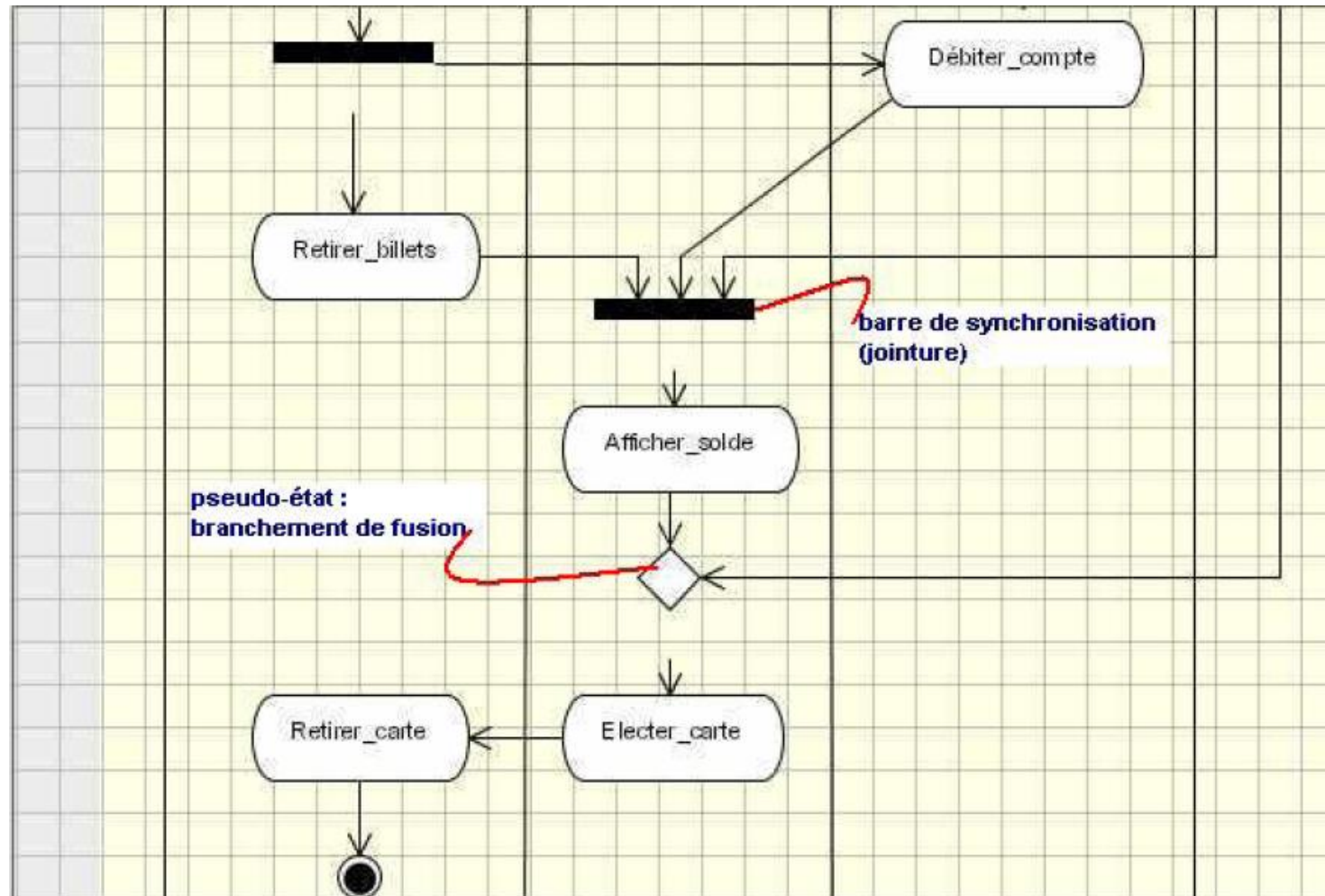
le client, le DAB et la banque.

■ Le processus débute, comme dans le diagramme d'états-transitions par un état initial représenté par un rond noir, et se termine de même par un état final représenté par un rond noir cerclé.

# Exemple de synthèse



# Exemple de synthèse

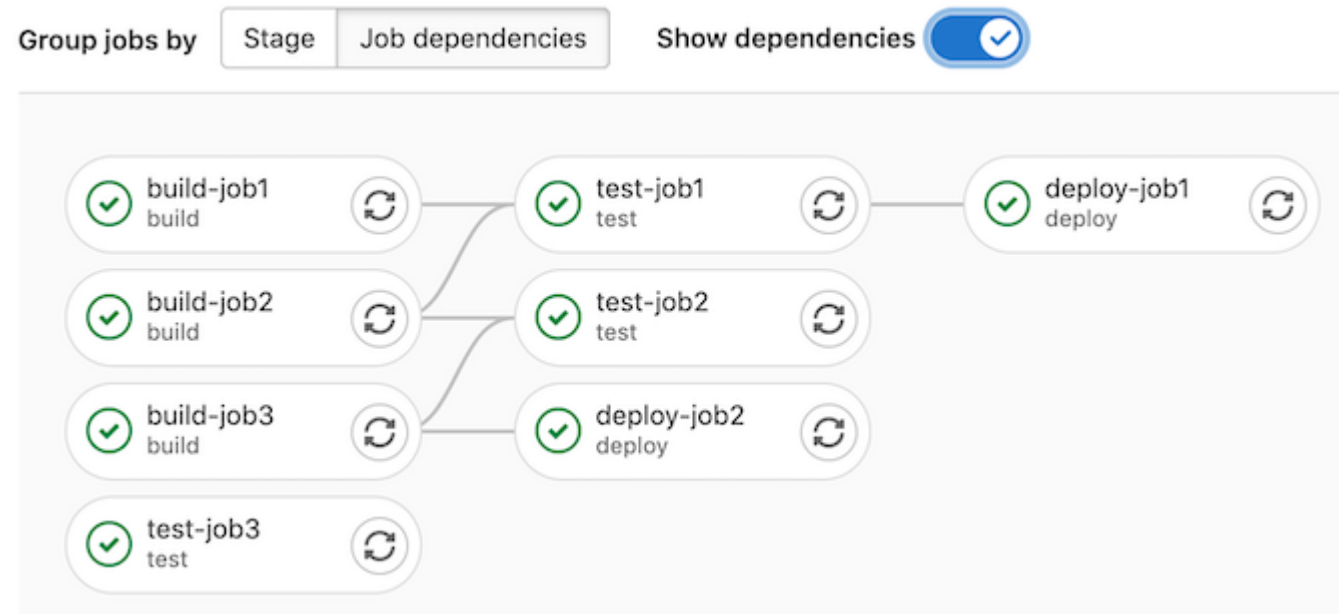


---

# L'AUTOMATISATION

---

- Pipelines
- Sont composé de jobs chapitrés en Stage
- Pilotés par **.gitlab-ci.yml** à la racine du repo



- Langage de formatage (XML, Json, Toml, Markdown, ...)
- Indentation sémantique !
- Cheatsheet : <https://quickref.me/yaml.html>

```
--- !clarkevans.com/^invoice
invoice: 34843
date : 2001-01-23
bill-to: &id001
  given : Chris
  family : Dumars
  address:
    lines: |
      458 Walkman Dr.
      Suite #292
    city : Royal Oak
    state : MI
    postal : 48046
ship-to: *id001
product:
  - sku : BL394D
    quantity : 4
    description : Basketball
    price : 450.00
  - sku : BL4438H
    quantity : 1
    description : Super Hoop
    price : 2392.00
tax : 251.42
total: 4443.52
comments: >
  Late afternoon is best.
  Backup contact is Nancy
  Billsmer @ 338-4338.
```

**SCALAR**

**COLLECTIONS**

**MULTI-LINE COLLECTIONS**

**LISTS/DICTIONARIES**

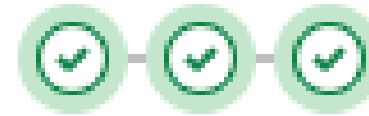
**MULTI-LINE FORMATTING**



- Possibilité de segmenté le workflow
- 5 stages par défaut :
  - .pre
  - build
  - test
  - deploy
  - .post

## Étapes

---



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

```
deploy:
  stage: deploy
  script: echo "Define your deployment script!"
  environment: production
```

## Noms réservés

- image
- services
- stages
- types
- before\_script
- after\_script
- variables
- cache
- include
- true
- false
- nil
- pages:deploy

## Les status

- failed
- warning
- pending
- running
- manual
- scheduled
- canceled
- success
- skipped
- created

- Instructions à exécuter comme sur un terminal

```
test:
```

```
script:
```

```
- vendor/bin/phpunit --configuration phpunit.xml --coverage-text --colors=never
```

---

# PAGES

---

## Pages

Avec GitLab Pages, vous pouvez héberger votre site Web statique directement sur votre dépôt GitLab. [En savoir plus.](#)

### ☒ Forcer le HTTPS (nécessite des certificats valides)

Lorsque cette option est activée, toutes les tentatives pour se rendre sur votre site Web via HTTP sont automatiquement redirigées vers HTTPS à l'aide d'une réponse avec le code d'état 301. Cette action nécessite un certificat valide pour tous les domaines. [En savoir plus.](#)

### ☒ Utiliser un domaine unique

Lorsque cette option est activée, un domaine unique est généré pour accéder aux pages.

Enregistrer les modifications

### Accès aux pages

<https://mycalculator-dawan-aldaitz-d21e316cb3d9021bf3e65dd240734a692eff.gitlab.io> 

Le contrôle d'accès est activé pour ce site Web des Pages ; seuls les utilisateurs autorisés seront en mesure d'y accéder. Pour rendre votre site Web accessible au public, accédez à **Paramètres > Général > Visibilité** dans votre projet et sélectionnez **Tout le monde** dans la section des pages. Veuillez consulter la [documentation](#) pour en savoir plus.

Domains (0)

Nouveau domaine

Vous n'avez actuellement aucun domaine personnalisé.

### Supprimer les pages

La suppression de pages les empêchera d'être exposées en externe.

Supprimer les pages

- *Déclarer Les stages : build, test, deploy*
- *Associé les bons jobs aux bons stages*
- *Créer un job de test associé au stage de test*

---

# ARTEFACTS

---

- Se définit au niveau du job.
- Les jobs peuvent utiliser les artefacts produits par les jobs précédents.
- Pas de partage entre les projets.
- Durée de vie de 30j (paramétrable).
- Utiliser « dependencies » pour stipuler quel job prend quel artefact



---

# LES TESTS

---

# Les tests

- Un **test** désigne une procédure de vérification partielle d'un système. Plus le nombre d'erreurs trouvé est important, plus il y a de chances qu'il y ait davantage d'erreurs dans le composant logiciel visé.
- Les tests de vérification ou de validation visent à s'assurer que ce système réagit de la façon prévue par ses concepteurs (spécifications) ou est conforme aux attentes du client l'ayant commandé (besoins), respectivement.

# Les types de test

- **Les tests unitaires** : Les tests unitaires consistent à tester individuellement les composants de l'application. On pourra ainsi valider la qualité du code et les performances d'un module.
- **Les tests d'intégration** : Ces tests sont exécutées pour valider l'intégration des différents modules entre eux et dans leur environnement exploitation définitif. Ils permettront de mettre en évidence des problèmes d'interfaces entre différents programmes.

# Les types de test - 2

- **Les tests fonctionnels** : Ces tests ont pour but de vérifier la conformité de l'application développée avec le cahier des charges initial. Ils sont donc basés sur les spécifications fonctionnelles et techniques.
- **Les tests de non-régression** : Les tests de non-régression permettent de vérifier que des modifications n'ont pas altérées le fonctionnement de l'application. L'utilisation d'outils de tests, dans ce domaine, permet de faciliter la mise en place de ce type de tests.

# Les types de test - 3

- **Les tests FrontEnd** : Les tests IHM ont pour but de vérifier que la charte graphique a été respectée tout au long du développement. Et que le code FrontEnd fonctionne également comme prévu.
- **Les tests de performance**
- **Les tests d'installation**

---

# MULTI-PIPELINES ?

---

Oui et non

## Fichier local

```
include: '.gitlab-ci-production.yml'
```

## Autre projet

```
include:  
- project: 'my-group/my-project'  
  file: '/templates/.gitlab-ci-template.yml'  
- project: 'my-group/my-subgroup/my-project-2'  
  file:  
    - '/templates/.builds.yml'  
    - '/templates/.tests.yml'
```

- *Séparer le build, le test et le deploy en 3 fichiers distincts :*
  - *Build.yml*
  - *Test.yml*
  - *Deploy.yml*



---

# VARIABLES

---

- CI\_COMMIT\_BRANCH : Nom de la branche
- CI\_DEFAULT\_BRANCH : Nom de la branche par défaut sur le projet
- CI\_PROJECT\_PATH : Nom du projet
- GITLAB\_USER\_NAME : Nom d'utilisateur du déclencheur du pipeline

[https://docs.gitlab.com/ee/ci/variables/predefined\\_variables.html](https://docs.gitlab.com/ee/ci/variables/predefined_variables.html)

---

# RULES

---

- if
- changes
- exists
- allow\_failure
- variables
- when

Déclenche le job si Les conditions sont remplies

```
job:
  script: echo "Hello, Rules!"
  rules:
    - if: $CI_MERGE_REQUEST_SOURCE_BRANCH_NAME =~ /^feature/ && $CI_MERGE_REQUEST_TARGET_BRANCH_NAME != $CI_DEFAULT_BRANCH
      when: never
    - if: $CI_MERGE_REQUEST_SOURCE_BRANCH_NAME =~ /^feature/
      when: manual
      allow_failure: true
    - if: $CI_MERGE_REQUEST_SOURCE_BRANCH_NAME
```

Déclenche le job si un changement a eu lieu

```
docker build:
  script: docker build -t my-image:$CI_COMMIT_REF_SLUG .
  rules:
    - if: $CI_PIPELINE_SOURCE == "merge_request_event"
      changes:
        - Dockerfile
      when: manual
      allow_failure: true
```

Déclenche le job si les éléments existent

```
job:  
  script: docker build -t my-image:$CI_COMMIT_REF_SLUG .  
  rules:  
    - exists:  
      - Dockerfile
```

Rend le job non bloquant

```
job:
  script: echo "Hello, Rules!"
  rules:
    - if: $CI_MERGE_REQUEST_TARGET_BRANCH_NAME == $CI_DEFAULT_BRANCH
      when: manual
      allow_failure: true
```



# VARIABLES

```
variables:
  DEPLOY_SITE: "https://example.com/"

deploy_job:
  stage: deploy
  script:
    - deploy-script --url $DEPLOY_SITE --path "/"
  environment: production

deploy_review_job:
  stage: deploy
  variables:
    REVIEW_PATH: "/review"
  script:
    - deploy-review-script --url $DEPLOY_SITE --path $REVIEW_PATH
  environment: production
```

# WHEN

```
cleanup_build_job:
  stage: cleanup_build
  script:
    - cleanup build when failed
  when: on_failure

test_job:
  stage: test
  script:
    - make test

deploy_job:
  stage: deploy
  script:
    - make deploy
  when: manual
  environment: production

cleanup_job:
  stage: cleanup
  script:
    - cleanup after jobs
  when: always
```

- `on_success` (default): Quand tous les précédents sont ok
- `on_failure`: Quand au moins un job a échoué (Sauf si `allow_failure`)
- `never`: Jamais
- `always`: Toujours
- `manual`: Quand le job est déclenché manuellement
- `delayed`: Pour instaurer un délai dans le déclenchement du job

- *Réaliser une page d'inaccessibilité qui doit être mis en ligne s'il y a la moindre erreur dans le pipeline*
  - *Ajouter eslint sur le projet*
    - *Le Monter en job*
    - *Tester*
  - *Puis rendre le job eslint facultatif*

---

# HÉRITAGE

---

```
.hidden_job:  
  script:  
    - run test
```



# EXTENDS

```
.tests:  
  script: rake test  
  stage: test  
  only:  
    refs:  
      - branches  
  
rspec:  
  extends: .tests  
  script: rake rspec  
  only:  
    variables:  
      - $RSPEC
```

---

# LES RUNNERS

---

- Délégation de commande
- Proxy
- Runner VS Exécuter
- A configurer dans Paramètres -> CI/CD -> Runners



# PROCESS

