

# Teamwork

Join the Git Cult

Romain Eggermont





# Summary

- Introduction
- Continuous Integration (CI)
- Going further
  - Work valuation
  - Git Hooks
  - & more



# Introduction

- In the three previous courses, you saw how to work with git and its ecosystem



# Continuous Integration

Problem: Several developers works together on the same project. How can I assure the code base is correct and viable?

Solution: Continuous Integration (CI)

Main goals: Lint and Test at each step of the project



# How does it work?

- Branches usually hooked. When a change is detected, a pipeline of multiple jobs is triggered
- CI tool answers with a status code to inform the git repository manager (GitLab / GitHub)



# Tools comparison

Tools	GitLab CI 	GitHub Actions 	Jenkins 	Travis CI 
Requirements	None / Runner (for self-hosted)	None	Server (self-hosted only)	\$\$\$
Pricing	Free	Free ( <a href="#">Limited</a> )	Free	\$\$\$
Features	++	+	+++	+++



# GitLab CI

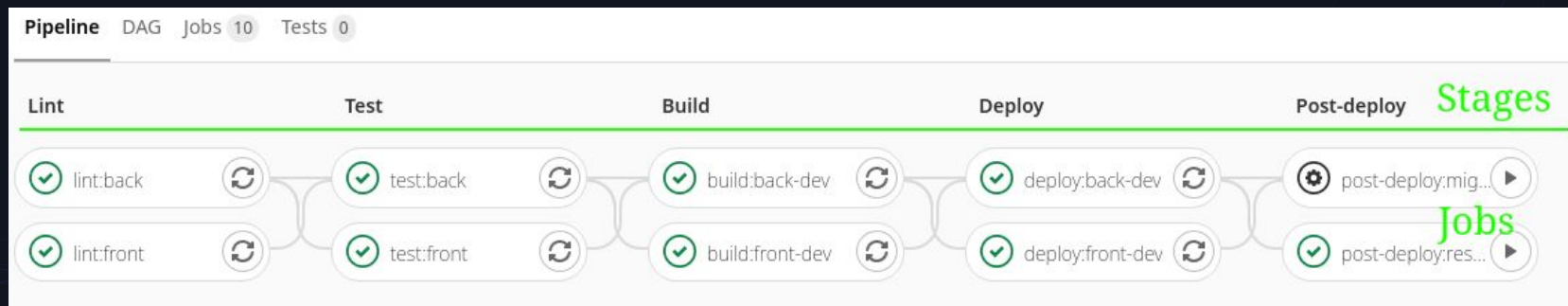
- Native support for GitLab, file written in YAML
- At each change, GitLab check for a `.gitlab-ci.yml` file, analyze it and run pipeline and jobs related to the changes made
- Based on Docker image: `image: node:lts`
- Test your yaml online at `/-/ci/lint`

```
1  image: node:lts
2
3  stages:
4  | - lint
5  | - test
6
7  code-lint: # Job n°1 declaration
8  | stage: lint
9  | only:
10 |   refs:
11 |     - master
12 |     - merge_requests
13 | scripts:
14 |   # The script
15
16 code-test: # Job n°2 declaration
17 | stage: test
18 | only:
19 |   refs:
20 |     - master
21 |     - merge_requests
22 | scripts:
23 |   # The script
24
```



# Stages and jobs

- Documentation for [stages](#) and [jobs](#)
- Define specific stages to order, separate and optionally parallelize jobs
- Jobs = single task







# GitHub Actions

- Native support for GitHub, file written in YAML
- At each change, GitHub check for a yaml file in `.github/workflows/`, analyze it and run pipeline and jobs related to the changes made
- CI results can be seen in `/actions`

```
1  name: CI
2
3  on:
4    push:
5      branches: [ main ]
6    pull_request:
7      branches: [ main ]
8
9  jobs:
10   test:
11     runs-on: ubuntu-latest
12     steps:
13       - uses: actions/checkout@v2
14
15       - name: Test check 1
16         run: echo "Test 1"
17
18       - name: Test check 2
19         run: echo "Test 2"
20
```



# Going further

The following slides are a non-exhaustive list of tips & tricks possible with Git

You can find a big list in the [Git book](#)



# Graphical log representation

- Following is an example, find the one that fit your tastes

- `git log --graph --pretty=format:'%Cred%h%Creset  
-%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold  
blue)<%an>%Creset' --abbrev-commit  
--date=relative`

- [Format cheatsheet](#)

- [Git Log documentation](#)

```
737f405 - (HEAD -> master, origin/master, origin/HEAD) Merge pull request #50 from scotthovestadt/dependabot/npm_and_yarn
* 33f850f - Bump lodash from 4.17.15 to 4.17.19 (11 months ago) <dependabot[bot]>
* 0204409 - Merge pull request #52 from renzosunico/feat-fidm.oidc.op.getMetadata-support (10 months ago) <Or Bar Yosef>

* 31191fe - Increment package version. (10 months ago) <jsunico>
* 3bd608c - Added getMetadata endpoint to fidm.oidc.op (10 months ago) <jsunico>

* 9111f18 - Merge pull request #48 from JordanShurmer/dep-versions (1 year, 6 months ago) <Or Bar Yosef>
* f3962c1 - Update panva/jose to jose and lodash version (from npm audit) (1 year, 6 months ago) <Jordan Shurmer>

* fcef170 - 3.0.1 (1 year, 6 months ago) <baryo>
* a07c995 - Merge pull request #45 from eliaivaman/master (1 year, 6 months ago) <Or Bar Yosef>

* adfa209 - revert package.json and package-lock.json (1 year, 6 months ago) <eliav.ma>
* 5ba53a3 - update authentication section in Policy object (1 year, 6 months ago) <eliav.ma>
* cc72afa - update authentication section in Policy object (1 year, 6 months ago) <eliav.ma>

/
* 105280d - updating major version (1 year, 9 months ago) <baryo>
* bf54a20 - Merge pull request #40 from scotthovestadt/sign-requests (1 year, 9 months ago) <iBaryo>

/
* ad55992 - (origin/sign-requests) update readme with jwt credentials (1 year, 9 months ago) <baryo>
* e91ee2b - type fix (1 year, 9 months ago) <baryo>
* f888f9a - refactor factories to signers (1 year, 9 months ago) <baryo>
* d0a7e6a - refactor setting credentials (1 year, 9 months ago) <baryo>
* 117fb0b - refactor setting credentials (1 year, 9 months ago) <baryo>
* 6305e07 - auth bearer requests (1 year, 9 months ago) <baryo>
* 5e55a9c - refactor to requests factories (1 year, 9 months ago) <baryo>
* 6a92020 - types cleanup (1 year, 9 months ago) <baryo>
* 4974590 - signing requests (1 year, 9 months ago) <baryo>

/
* 1724970 - 2.0.43 (1 year, 11 months ago) <baryo>
* 054fb91 - 2.0.42 (1 year, 11 months ago) <baryo>
* 5ecd10d - Merge pull request #38 from BaramD/master (1 year, 11 months ago) <iBaryo>

/
* ac63534 - Merge branch 'develop' (1 year, 11 months ago) <Dmitriy Baram>
* dd55340 - Merge branch 'us/87910-add-global-site-properties' into develop (1 year, 11 months ago) <Dmitriy Baram>
* fcc3ada - .vsd files removed (1 year, 11 months ago) <Dmitriy Baram>
* b7fd0a1 - Merge branch 'develop' (1 year, 11 months ago) <Dmitriy Baram>

* e8d74d7 - Merge branch 'us/87910-add-global-site-properties' into develop (1 year, 11 months ago) <Dmitriy Baram>

* 9917d46 - Some files removed (1 year, 11 months ago) <Dmitriy Baram>
* f2b566a - Merge pull request #2 from BaramD/develop (1 year, 11 months ago) <BaramD>
```



# Work valuation

- Thanks to Git, you can easily make reports of work made by the team
- Here, we just give two commit hashes to a custom script that generates the stats

```
> ./scripts/release/review_stats.sh 846d29d cf5956a
Number of commits: 195
Summary of changes: 213 files changed, 1367 insertions(+), 127178 deletions(-)
Number or merged PRs: 5
```



# Submodules

- Submodules is the package manager for git repositories

- Syntax: `git submodule add [-b <branch>] <repository> [<path>]`

- [Git Submodule documentation](#)



# Git Hooks

- Run custom scripts when specific important git actions occur.
- Two types:
  - Client-side hooks: pre-commit, post-commit, pre-push, ...
  - Server-side hooks: pre-receive, post-receive, update, ...
- Skip hooks with `--no-verify` option
- To be edited in the `.git/hooks` folder, where you can find hooks samples
- Standard: Put them in `.githubhooks` folder to version it and run `git config --local core.hooksPath .githubhooks/`
- [Git Hooks Documentation](#)



# Repositories synchronisation

- The goal is to save code at multiple sources (GitHub + GitLab for example)
- Usually works as a post-receive hook, as seen before.
- Can be made manually as a hook, or thanks to GitLab or GitHub
- [GitLab mirroring documentation](#)