27 février 2020

PHP en continu grâce à GitLab

Sébastien Ballangé

ConFoo.CA
DEVELOPER CONFERENCE

paystone

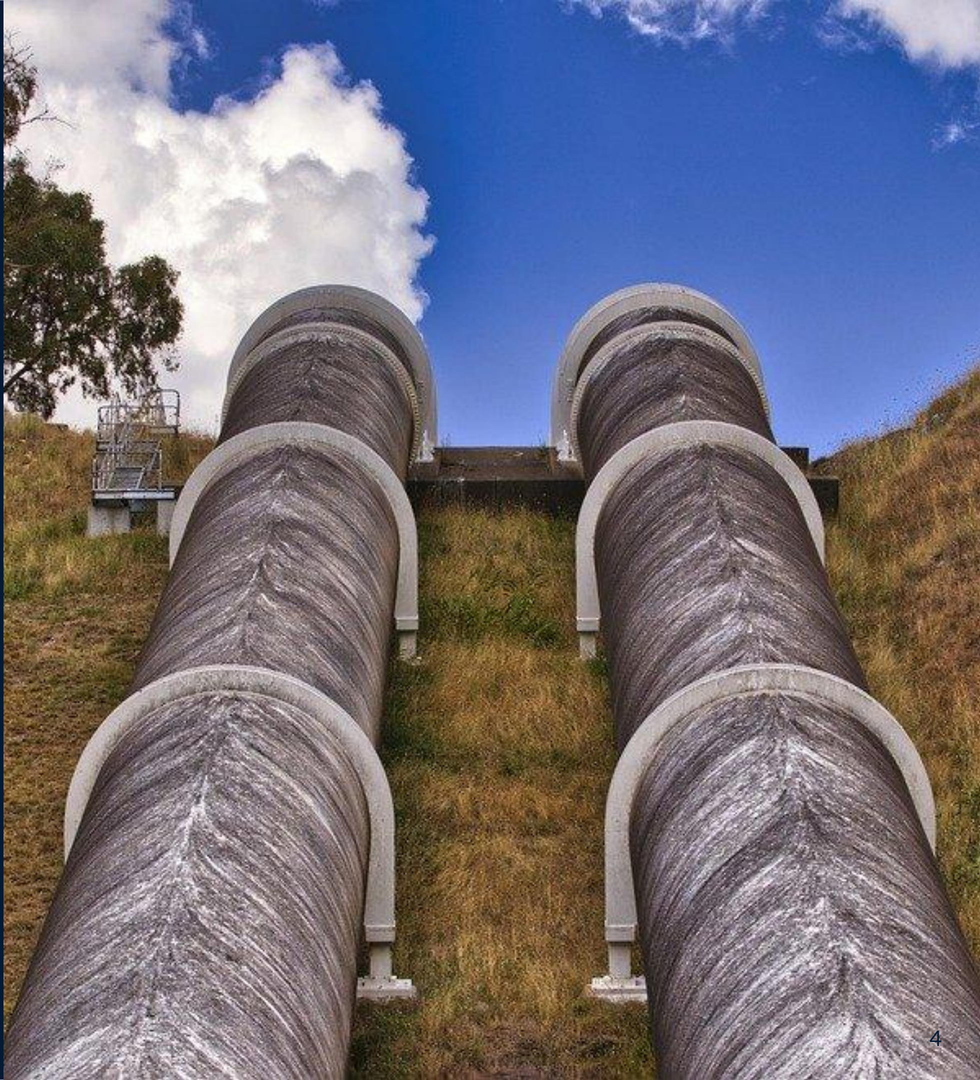
GitLab ?

- À l’origine un “clone” open-source de GitHub
- Hébergé à l’interne ou disponible comme service via gitlab.com
- Gratuit ou payant selon les versions
- “Plateforme DevOps” (Git, CI/CD, gestion de projet, …)

CI/CD?

- CI: Continuous Integration (Intégration en continu)
- CD: Continuous Delivery (Livraison en continu)

Les “pipelines” de GitLab



.gitlab-ci.yml

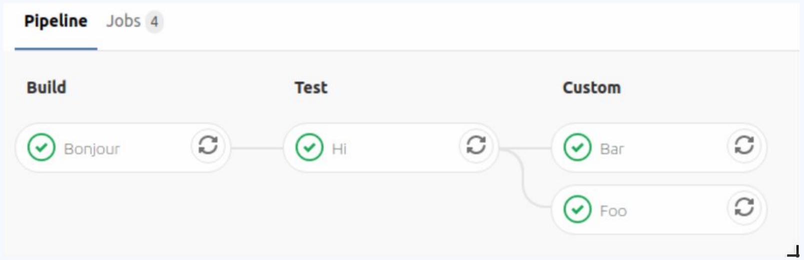
https://docs.gitlab.com/ee/ci/README.html

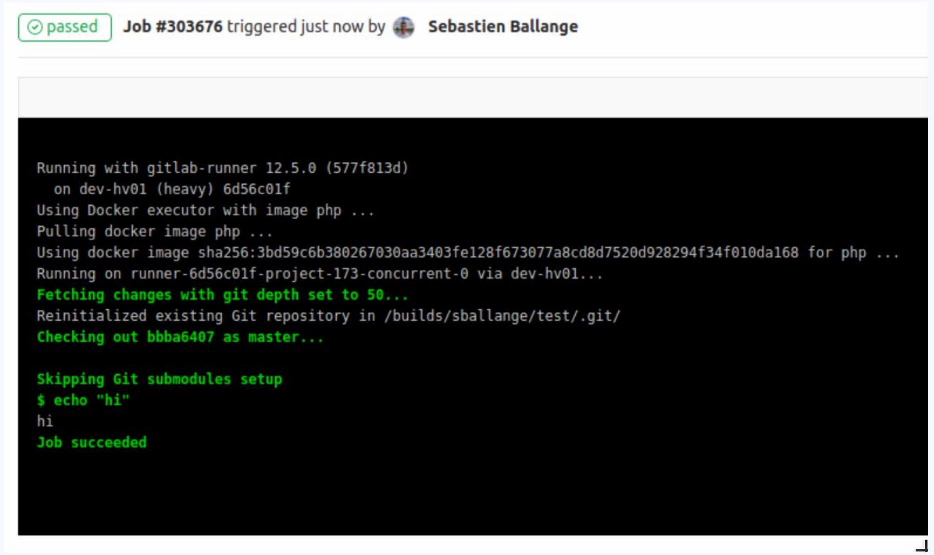
```
Bonjour:
 script:
   - echo "hello world"
```

Grouper les étapes

```yaml
stages:
  - build
  - test
  - custom

Bonjour:
 stage: build
 script:
   - echo "bonjour"

Hi:
 stage: test
 script:
   - echo "hi"

Foo:
 stage: custom
 script:
   - echo "foo"

Bar:
 stage: custom
 script:
   - echo "bar"
```

Grouper les étapes





Job #303676 triggered just now by Sebastien Ballange

```
Running with gitlab-runner 12.5.0 (577f813d)
  on dev-hv01 (heavy) 6d56c01f
Using Docker executor with image php ...
Pulling docker image php ...
Using docker image sha256:3bd59c6b380267030aa3403fe128f673077a8cd8d7520d928294f34f010da168 for php ...
Running on runner-6d56c01f-project-173-concurrent-0 via dev-hv01...
Fetching changes with git depth set to 50...
Reinitialized existing Git repository in /builds/sballange/test/.git/
Checking out bbba6407 as master...

Skipping Git submodules setup
$ echo "hi"
hi
Job succeeded
```

Composer + PHPUnit

```yaml
image: php:7.4

stages:
  - build
  - test

Composer:
  stage: build
  script:
    - composer install --optimize-autoloader

PHPUnit:
  stage: test
  script:
    - vendor/bin/phpunit --log-junit=phpunit-report.xml
  artifacts:
    when: always
    expire_in: 2 weeks
    reports:
      junit: phpunit-report.xml
```

Composer + PHPUnit (2)

```yaml
image: php:7.4

PHPUnit:
  stage: test
  before_script:
    - composer install --optimize-autoloader
  after_script:
    - echo "Tests terminés"
  script:
    - vendor/bin/phpunit --log-junit=phpunit-report.xml

  artifacts:
    when: always
    expire_in: 2 weeks
    reports:
      junit: phpunit-report.xml
```

Code Coverage

```
image: php:7.4

PHPUnit:
  stage: test
  script:
    - composer install --optimize-autoloader
    - vendor/bin/phpunit --log-junit=phpunit-report.xml --coverage-text
--colors=never

  coverage: '/^\s*Lines:\s*\d+.\d+\%/'
  artifacts:
    when: always
    expire_in: 2 weeks
    reports:
      junit: phpunit-report.xml
```

Cache

```yaml
image: php:7.4

PHPUnit:
  stage: test
  script:
    - composer install --optimize-autoloader
    - vendor/bin/phpunit --log-junit=phpunit-report.xml

  artifacts:
#     [...]

cache:
  key: ${CI_COMMIT_REF_SLUG}
  paths:
    - vendor/
    - node_modules/
```

Limiter les déclenchements

```yaml
PHPUnit:
  only:
    - merge_requests
    - tags
```

```yaml
PHPUnit:
  only:
    - branches
# ou
  only:
    - master
# ou
  only:
    - /^(master|release).*/
```

```yaml
Deploiement:
  only:
    - master
  when: manual
```

Options intéressantes

- interruptible: true
- allow_failure: true
- dependencies: []

Réduire la duplication

Plusieurs versions de PHP ?

```
.phpunit:
  stage: test
  script:
    - composer install
    - vendor/bin/phpunit

PHPUnit (PHP 7.2):
  extends: .phpunit
  image: php:7.2

PHPUnit (PHP 7.3):
  extends: .phpunit
  image: php:7.3

PHPUnit (PHP 7.4):
  extends: .phpunit
  image: php:7.4
```

YAML “Anchors”

```yaml
.phpunit: &base_phpunit
  stage: test
  script:
    - composer install
    - vendor/bin/phpunit

PHPUnit (PHP 7.2):
  <<: *base_phpunit
  image: php:7.2

PHPUnit (PHP 7.3):
  <<: *base_phpunit
  image: php:7.3

PHPUnit (PHP 7.4):
  <<: *base_phpunit
  image: php:7.4
```

Partager la définition du pipeline entre projets

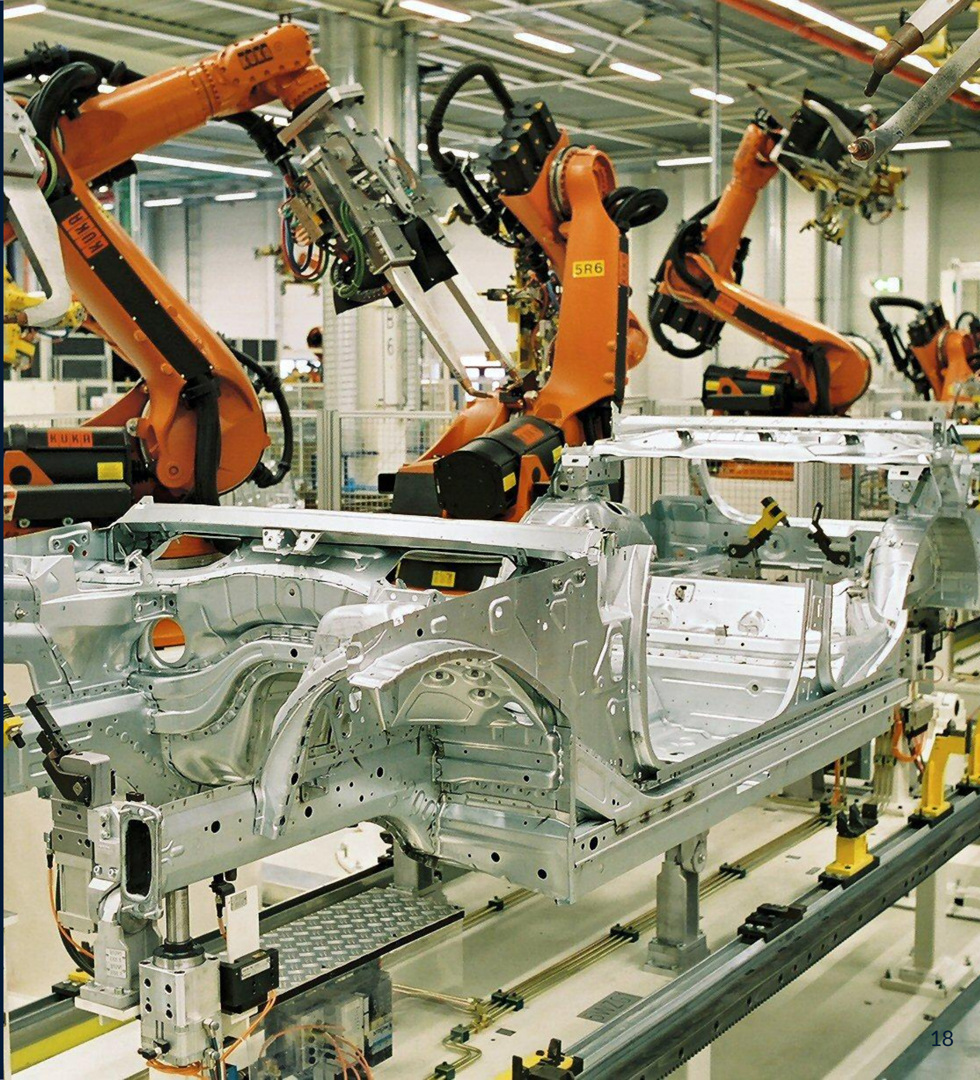
```yaml
# my-company/the-good-team/cool-project/.gitlab-ci.yml

include:
  - project: my-company/common/gitlab-templates
    file: phpunit.yml

Autre chose:
  stage: test
  script:
    - echo “The Good Place”
```

```yaml
# my-company/common/gitlab-templates/phpunit.yml

PHPUnit:
  stage: test
  script:
    - composer install
    - # [...]
    - vendor/bin/phpunit
```

```yaml
# my-company/the-other-team/boring-project/.gitlab-ci.yml

include:
  - project: my-company/common/gitlab-templates
    file: phpunit.yml

Autre chose:
  stage: test
  script:
    - echo “The Bad Place”
```

Automatiser!

Automatiser!



Analyse statique: SensioLabs Security Checker

```
SensioLabs Security Checker:
  stage: .post
  image: composer:latest
  dependencies: []
  rules:
    - exists:
      - composer.lock
  before_script:
    - export PATH="/tmp/vendor/bin:$PATH"
    - composer global require sensiolabs/security-checker
  script:
    - security-checker security:check
```

Analyse statique: librairies désuètes

```
Composer Outdated Packages:
  stage: .post
  image: composer:latest
  only:
    - /^(master|release).*/

  script:
    - composer outdated --strict
```

Analyse statique: standards de code

```yaml
PHP_CodeSniffer:
  stage: .post
  image: composer:latest
  dependencies: []
  before_script:
    - export PATH="/tmp/vendor/bin:$PATH"
    - composer global require squizlabs/php_codesniffer
  script:
    - git clean -fdx
    - phpcs --standard=PSR12 -p --ignore=*.js,vendor/ --report=summary ./
```

Analyse statique: Code Climate (GitLab)

```
include:
    - template: Code-Quality.gitlab-ci.yml

code_quality:
  stage: .post
  allow_failure: true
  only:
    - /^(master|release|feature\/).*/ # Run Code Climate for master, release and
feature branches
```

Mise à jour automatique d’une branche

```
Update Release Branch:
  stage: .post
  image: $CI_REGISTRY/common/php:latest # Image avec clé SSH autorisée dans GitLab
  allow_failure: true # ou false pour échouer en cas de conflits
  dependencies: [] # ne pas copier les artéfacts des autres étapes
  only:
    - master

  before_script:
    - git config user.email "$GITLAB_USER_EMAIL"
    - git config user.name "$GITLAB_USER_NAME"

  script:
    - git fetch --prune
    - git checkout origin/release
    - git merge origin/$CI_COMMIT_REF_NAME
    - git remote set-url origin git@$CI_SERVER_HOST:$CI_PROJECT_PATH.git
    - git push origin HEAD:release
```

Formatage automatique du code

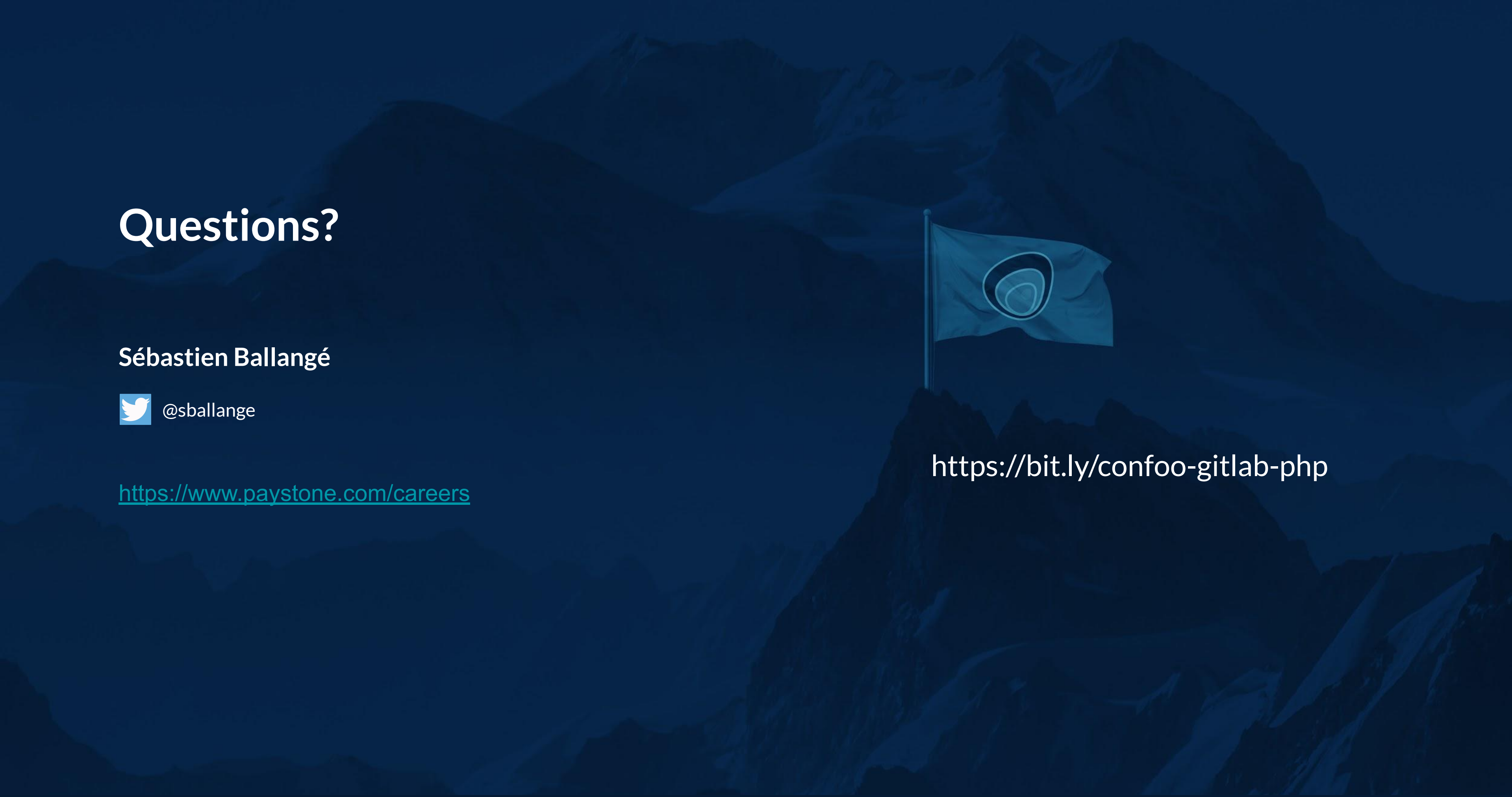
```
Fix Code Style:
  stage: .post
  image: $CI_REGISTRY/common/php:latest # Image avec clé SSH autorisée dans GitLab

  before_script:
    - git config user.email "$GITLAB_USER_EMAIL"
    - git config user.name "$GITLAB_USER_NAME"
    - composer global require squizlabs/php_codesniffer
    - git --version # --push-option nécessite Git 2.10+

  script:
    - git clean -fdx
    - phpcbf --standard=PSR12 -p --ignore=*.js,vendor/ ./ || echo "Il reste des erreurs"
    - git commit -a -m "[GitLab 🤖] Fix PSR 12 Violations"
    - git remote set-url origin git@$CI_SERVER_HOST:$CI_PROJECT_PATH.git
    - git push --push-option=merge_request.create \
          --push-option=merge_request.target=$CI_COMMIT_REF_NAME \
          --push-option=merge_request.remove_source_branch \
          origin \
          HEAD:fix-code-standard-$CI_COMMIT_SHORT_SHA
```

Docker build

```
Update Docker Container:
  stage: .post
  image: docker:stable
  interruptible: true
  variables:
    DOCKER_DRIVER: overlay2
  only:
    refs:
      - master
    changes:
      - Dockerfile
      - docker/*
  services:
    - docker:stable-dind

  script:
    - docker login -u $CI_REGISTRY_USER -p $CI_REGISTRY_PASSWORD $CI_REGISTRY
    - docker build --pull --tag $CI_REGISTRY_IMAGE:$CI_COMMIT_SHORT_SHA --tag 
$CI_REGISTRY_IMAGE:latest .
    - docker push $CI_REGISTRY_IMAGE:$CI_COMMIT_SHORT_SHA
    - docker push $CI_REGISTRY_IMAGE:latest
```

Déploiement automatique

```yaml
.deploy:
  stage: deploy
  image: $CI_REGISTRY/common/deploy:latest # Image avec config par défaut (clés SSH, …)
  only: master
  script:
    - echo "Deploying to $TARGET_SERVER..."
    - composer install --optimize-autoloader --no-dev
    - yarn encore production
    - rsync -azh --delete --exclude .git --exclude var .$TARGET_SERVER:/var/www/
    - ssh myuser@$TARGET_SERVER "/var/www/bin/console cache:clear"

Deploy Staging:
  extends: .deploy
  variables:
    TARGET_SERVER: staging.office.example.ca
  environment:
    name: staging
    url: https://staging.example.ca

Deploy Production:
  extends: .deploy
  when: manual
  variables:
    TARGET_SERVER: production.example.ca
  environment:
    name: production
    url: https://www.example.ca
```



Questions?

Sébastien Ballangé

@sballange

https://www.paystone.com/careers

https://bit.ly/confoo-gitlab-php

Crédits Photos

- https://pixabay.com/photos/pipes-hydroelectricity-pipeline-4502828/
- https://commons.wikimedia.org/wiki/File:Bike_share_clutter_Beijing.jpg
- https://commons.wikimedia.org/wiki/File:KUKA_Industrial_Robots_IR.jpg