

UNIVERSITE E FRANCHE-COMTE

Fabrice Bouquet et Anthony Dugois

UNIVERSITE FRANCHE-COMTE

2023 -- 2024

Université de Franche-Comté Institut FEMTO-ST

DISC, Besançon, France







### Présentation de Unitestor

Un robot qui se déplace sur une planète lointaine...

https://disc.univ-fcomte.fr/m2gl-webRobot/Accueil.php

### Composants

- une horloge interne
- une batterie
- un panneau solaire
- un GPS
- un capteur de sol

#### **Fonctionnalités**

- déplacement : en fonction du sol et de la distance, plus ou moins d'énergie sera consommée
- rechargement de la batterie : le panneau solaire recharge la batterie de manière constante



# Architecture du projet UniTestor





- A télécharger sur Moodle : "UniTestor\_phpUnit.zip"
  - Source/Vendor/Unitestor : sources du projet
    - Clock.php
    - Coordinates.php
    - LandSendor.php
    - Robot.php
    - Vector.php
  - Test/Vendor/Unitestor : tests unitaires manuels
    - CoordinatesTest.php



- Chaîne d'intégration continue (https://docs.gitlab.com/ee/ci/)
- Un pipeline est composé de stages.
- Un stage est composé de jobs.
- Le runner Gitlab exécute chaque job dans un environnement d'exécution prédéfini (shell, Docker, etc.).

On veut exécuter les tests unitaires lors d'un *push* sur le dépôt.





- 1. Récupérer les sources du projet "UniTestor\_phpUnit.zip"
- 2. Créer un dépôt de test sur Gitlab
- 3. Pousser les sources du projet sur le dépôt



- 1. Créer un fichier .gitlab-ci.yml.
- 2. Ajouter un job :

```
1 test:
2 image: php:8.2—apache
3 script:
4 — phpunit tests
```

Par défaut, l'image Docker php:8.2-apache ne comprend pas PHPUnit. On va utiliser un script d'initialisation pour s'assurer que PHPUnit est bien présent.



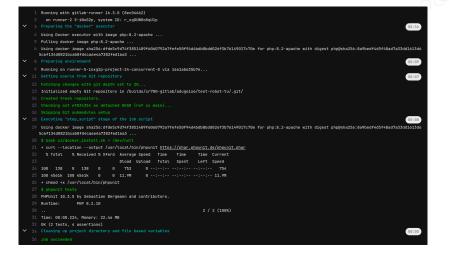
```
Au début de .gitlab-ci.yml :
default:
  before_script:
    — bash ci/docker_install.sh > /dev/null
Puis créer le fichier ci/docker_install.sh :
#!/bin/bash
[[ ! -e /.dockerenv ]] \&\& exit 0
set -xe
curl — location — output / usr/local/bin/phpunit \
    "https://phar.phpunit.de/phpunit.phar"
chmod +x /usr/local/bin/phpunit
```

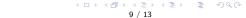
9



- Pousser les nouveaux fichiers
- S'assurer que les jobs s'exécutent correctement en allant dans Build, puis Jobs







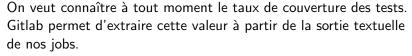




- Refaire la même chose, mais cette fois pour les tests selenium
- tester sur le robot : https:
  //disc.univ-fcomte.fr/m2gl-webRobot/Accueil.php
- L'image à utiliser : docker.io/markhobson/maven-chrome:jdk-8
- command: mvn -f /usr/src/app/pom.xml test



### Calculer/afficher la couverture



Tout d'abord, on commence par calculer le taux de couverture avec PHPUnit. Pour cela, installer l'extension PHP Xdebug. Dans ci/docker\_install.sh ajouter :

```
1 ...
```

- 2 pecl install xdebug
- 8 docker—php—ext—enable xdebug



# Calculer/afficher la couverture

```
Puis, dans .gitlab-ci.yml :
123456789
    test:
      image: php:8.2 — apache
      script:

    XDEBUG_MODE=coverage phpunit tests/units

            -coverage-filter tests/units
            -do-not-cache-result
            —log-junit phpunit-report.xml
            -coverage-cobertura phpunit-coverage.xml
            -coverage-text -colors=never
10
      artifacts:
11
        when: always
12
        reports:
13
          junit: phpunit-report.xml
14
           coverage_report:
15
             coverage_format: cobertura
16
             path: phpunit-coverage.xml
17
      coverage: '/^\s*Lines:\s*\d+.\d+\%/'
```



## Calculer/afficher la couverture

La ligne de commande PHPUnit permet de générer les rapports de couverture sous différents formats, plus tard utilisés par Gitlab. La commande artifacts indique à Gitlab les fichiers à sauvegarder en tant qu'artéfacts des jobs (autrement, les rapports

sauvegarder en tant qu'artéfacts des jobs (autrement, les rapports seraient simplement détruits à la fin du job).

Enfin, la commande coverage indique à Gitlab comment extraire le taux de couverture depuis la sortie textuelle du job.

Après plusieurs commits, on peut visualiser l'évolution du taux de couverture en allant dans Analyze puis Repository analytics.

https://docs.gitlab.com/ee/ci/testing/code\_coverage.html https://docs.gitlab.com/ee/ci/testing/test\_coverage\_ visualization.html

