

Automatizzare e Ottimizzare CI/CD



PUG Torino

- [PHP User Group Torino](#) is a group of web developers interested in the PHP language (and not only)
- We are part of [GrUSP](#) association
- On [meetup.com](#) we are about 621 members
- We have also a [mailing list](#) with more than 100 members
- [Toolbox Coworking](#) is sponsoring the group



TOOLBOX
COWORKING





Daniele Barbaro

Tech Lead

Classe 84, workaholic, inge, in eterna lotta per la conquista del mondo.

Da ormai 14 anni ha fatto del PHP la sua arma di conquista.



<https://daniele.barbaro.online>



Perché siamo qua?!



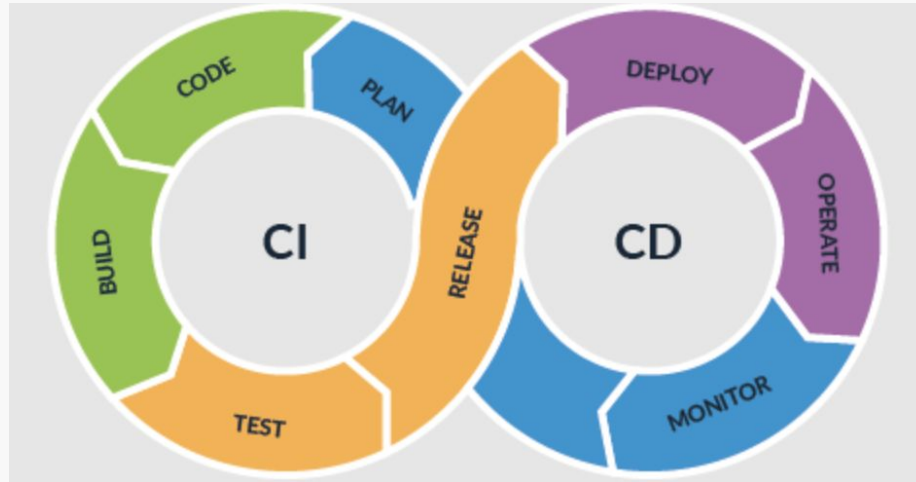


Perché siamo qua?!

- capire cosa significa CI/CD
- capire perché è importante usare questi strumenti
- capire gli strumenti giusti per affrontare ogni progetto
- capire l'importanza di automatizzare, ottimizzare processi e testare
- capire come affrontare un refactor



Continuous Integration (CI) Continuous Deployment/Delivery (CD)





Continuous Integration (CI)

Processo di test automatico e integrazione continua delle modifiche al codice

- Ogni modifica al codice viene testata automaticamente.
- Le modifiche vengono unite nel branch principale se superano i test.





Continuous Deployment/Delivery (CD)

Processo di distribuzione automatica delle applicazioni negli ambienti di produzione.

- Ogni modifica che supera i test di integrazione continua viene automaticamente distribuita.
- Elimina le necessità di intervento manuale per le distribuzioni.





Riepilogando:

- **CI** garantisce che li eseguiamo
- **CD** lo rilascia in modo affidabile

In generale, i **TEST** garantiscono il funzionamento del nostro software in tutte le fasi della pipeline



Perchè?!



Perchè?!

- Ridurre drasticamente l'impatto umano nelle azione ripetitive
- Migliora la qualità del codice.
- Individuazione rapida dei problemi.
- Rilasciare nuove funzionalità rapidamente.
- Migliora la reattività alle richieste del mercato





In 3 parole:

- **Consistenza** - Sempre deployabile
- **Efficienza** - No azioni ripetitive manuali
- **Qualità** - Sempre testato



Workflow /



<https://github.com/marketplace>

Pipeline



<https://gitlab.com/pipeline-components>





Workflow

è una serie di job definiti in un file di configurazione yml che definisce cosa deve accadere quando si verifica un determinato evento nel repository.

- **Events:** Questi sono i trigger che attivano il workflow. Possono essere eventi come un push o una pull request ecc.
(<https://docs.github.com/en/actions/using-workflows/events-that-trigger-workflows>)
- **Job:** E' un insieme di steps che vengono eseguiti su una macchina virtuale o su un container. Un job può includere comandi di shell, script
- **Steps:** Sono le singole istruzioni all'interno di un job. Ogni passo può eseguire un comando o utilizzare un'azione.

```
name: CI

on: [push]

jobs:
  ci-test:
    runs-on: ubuntu-latest

    steps:
      - name: "Say Hello"
        run: echo "Hello
Daniele!"
```



Workflow

```
name: tests

on:
  push:
    branches:
      - master
      - develop
  pull_request:
    branches:
      - master

jobs:
  php-tests:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v3

      - name: Install PHP
        uses: "shivammathur/setup-php@v2"

      - name: Run composer
        run: composer install --prefer-dist --no-progress --no-ansi --no-
interaction
      - name: Run unit tests
        run: vendor/bin/phpunit
```





Code Quality - Php Stan

PHPStan esamina l'intero codice alla ricerca di bug evidenti e insidiosi, anche in quelle istruzioni if raramente eseguite che certamente non sono coperte dai test.

<https://phpstan.org/try>





Code Quality



```
- name: Run static analysis checks
  run: |
    vendor/bin/phpstan analyse src --error-format=checkstyle | cs2pr
```

*cs2pr: Annotate a Pull Request based on a Checkstyle XML

<https://github.com/staabm/annotate-pull-request-from-checkstyle>



Code Quality

27

28 + `private function ensureIsValidEmail($email): void`

✖ Check failure on line 28 in src/Email.php



GitHub Actions / php-tests

Method Danielebarbaro\CiCdSample\Email::ensureIsValidEmail() has parameter

29

{





Code Linting - Php cs

PHP CS Fixer analizza l'intero codice alla ricerca di problemi di stile e di formattazione, garantendo che il tuo codice rispetti gli standard stabiliti.

<https://cs.symfony.com/>





Code Style



```
- name: Check code style
  run: |
    vendor/bin/php-cs-fixer fix src --dry-run --format=checkstyle | cs2pr
```



Code Style Workflow

```
name: Check & fix styling

on: [push]

jobs:
  php-cs-fixer:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v2
        with:
          ref: ${ github.head_ref }

      - name: Run PHP CS Fixer
        uses: docker://oskarstark/php-cs-fixer-ga
        with:
          args: --config=.php_cs.dist.php --allow-risky=yes

      - name: Commit changes
        uses: stefanzweifel/git-auto-commit-action@v4
        with:
          commit_message: Fix styling
```



Test Workflow



```
name: run-tests

on:
  push:
    branches:
      - master
      - develop
  pull_request:
    branches:
      - master

jobs:
  test:
    runs-on: ${ matrix.os }

    strategy:
      fail-fast: true
      matrix:
        os: [ubuntu-latest, windows-latest]
        php: [8.3, 8.2]
        laravel: [ '10.*', '11.*' ]
        stability: [prefer-lowest, prefer-stable]
        include:
          - laravel: 10.*
            testbench: 8.*
          - laravel: 11.*
            testbench: 9.*

    name: P${{ matrix.php }} - L${{ matrix.laravel }} - ${{ matrix.stability }} - ${{ matrix.os }}

    steps:
      - name: Checkout code
        uses: actions/checkout@v3

      - name: Setup PHP
        uses: shivammathur/setup-php@v2
        with:
          php-version: ${{ matrix.php }}
          extensions: dom, curl, libxml, mbstring, zip, pcntl, pdo, sqlite, pdo_sqlite, bcmath, soap, intl, gd, exif, iconv, imagick, fileinfo, xdebug
          coverage: xdebug

      - name: Setup problem matchers
        run: |
          echo "::add-matcher::${{ runner.tool_cache }}/php.json"
          echo "::add-matcher::${{ runner.tool_cache }}/phpunit.json"

      - name: Install Matrix dependencies
        run: |
          composer require "laravel/framework:${{ matrix.laravel }}" "orchestra/testbench:${{ matrix.testbench }}" --no-interaction --no-update
          composer update --${{ matrix.stability }} --prefer-dist --no-interaction

      - name: Execute tests
        run: vendor/bin/phpunit
```













Gitlab Comparison

```
phpunit:
  stage: test
  image: registry.gitlab.com/pipeline-components/phpunit:latest
  variables:
    OS: "ubuntu-latest"
    LARAVEL_VERSION: "11.*"
    STABILITY: "prefer-stable"
    TESTBENCH: "9.*"
  script:
    - curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=composer
    - composer install
    - composer require "laravel/framework:$LARAVEL_VERSION" "orchestra/testbench:$TESTBENCH" --no-interaction
  --no-update
    - composer update --$STABILITY --prefer-dist --no-interaction
    - ./vendor/bin/phpunit --no-coverage

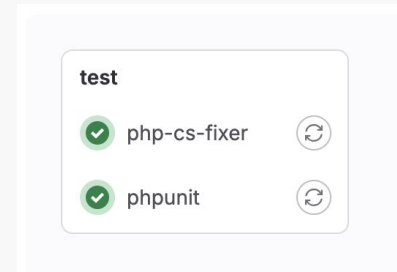
php-cs-fixer:
  stage: test
  image: registry.gitlab.com/pipeline-components/php-cs-fixer:latest
  script:
    - curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=composer
    - composer install
    - ./vendor/bin/php-cs-fixer fix src --dry-run --diff --verbose --show-progress=dots
```



GitHub

<div> All checks have passed Hide all checks</div> <div>1 skipped and 33 successful checks</div>		
	dependabot-auto-merge / dependabot (pull_request_target) Skipped	Details
	 Check & fix styling / php-cs-fixer (push) Successful in 5s	Details
	 run-tests / P8.3 - L10.* - prefer-lowest - ubuntu-latest (pull_request) Successful in 19s	Details
	 run-tests / P8.3 - L10.* - prefer-lowest - ubuntu-latest (push) Successful in 20s	Details
	 run-tests / P8.3 - L10.* - prefer-stable - ubuntu-latest (pull_request) Successful in 16s	Details
	 run-tests / P8.3 - L10.* - prefer-stable - ubuntu-latest (push) Successful in 13s	Details

Gitlab





Demo Time





Refactor & Pipeline

Rector analizza l'intero codice e applica automaticamente refactoring e aggiornamenti, migliorando la qualità del codice e assicurando che sia sempre aggiornato con le migliori pratiche.

Ti aiuta con:

- aggiornamenti PHP,
- aggiornamenti di framework,
- miglioramento della qualità del codice.





Workflow Ninja Mode

Oltre i semplici controlli di base:

- Controlli della licenze
- Creazione di risorse frontend (esempio build npm Tailwind).
- Integrazione Docker in CI.

Automazione della distribuzione:

- Tag e rilascio del codice.
- Utilizzo scripts GitHub per rilasci automatizzati.
- Caricamento degli artefatti della build.



Workflow Ninja Mode

- expressions
- contexts
- events



<https://docs.github.com/en/actions/learn-github-actions/expressions>

<https://docs.github.com/en/actions/learn-github-actions/contexts>

<https://docs.github.com/en/actions/using-workflows/events-that-trigger-workflows>

```
name: Example Workflow

on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Run step if condition is true
        if: github.event_name == 'push'
        run: echo "This step runs only on push events to the main
branch."

      - name: Always run this step
        run: echo "This step always runs."
```



Tools

App/Src:

- <https://github.com/PHP-CS-Fixer/PHP-CS-Fixer>

Template:

- <https://github.com/VincentLanglet/Twig-CS-Fixer>
- <https://github.com/bdelespierre/laravel-blade-linter>

Analysis:

- <https://github.com/phpstan/phpstan>

Refactor:

- <https://github.com/rectorphp/rector>
- <https://github.com/driftingly/rector-laravel/>



Grazie!



<https://daniele.barbaro.online>

Repo:

- <https://github.com/danielebarbaro/laravel-vat-eu-validator>
- <https://github.com/danielebarbaro/ci-cd-sample>

