

SEPTEMBER 15 - 16, 2022

# API PLATFORM CONFERENCE

LILLE (FRANCE) & ONLINE

# Tests avancés



**OLIVIER DOUCET**

CO-FONDATEUR

@ **OXEVA**

@ezameku

# QUI SUIS-JE ?

## **OLIVIER DOUCET**

\_ Cofondateur et directeur général d'Oxeva depuis 2005

\_ Côté dev :

Premier code sur PHP 4

Contributeur open source à de nombreux projets : PHP, MySQL, HAProxy...





QUE FAIT OXEVA ?

# SOLUTIONS D'HÉBERGEMENT INFOGÉRÉ EN CLOUD



Libérez-vous de vos problématiques d'hébergement et concentrez-vous uniquement sur votre activité.

# nua•ge

# DE QUOI VA-T-ON PARLER ?

- Tester une API distante, une spec OpenAPI et les ACL
- Tour d'horizon d'outils
- Une CI/CD efficace et lisible



TESTER, TESTER,  
**TESTER**

# TESTER L'INTESTABLE

## TESTER UNE API DISTANTE ... SANS L'APPELER

### Prérequis

- \_ Utiliser HttpClient
- \_ Encapsuler les appels dans une class MonApi, qui utilise une HttpClientInterface

### Solution

- \_ Déclarer un scoped client
- \_ Ajouter une classe « AbstractApiMock »
- \_ Utiliser un décorateur en test

```
1  <?php
2
3  namespace App\Tests\Mock;
4
5  use Symfony\Component\HttpClient\MockHttpClient;
6  use Symfony\Component\HttpClient\Response\MockResponse;
7  use Symfony\Component\HttpFoundation\Response;
8
9  final class AbstractApiMock extends MockHttpClient
10 {
11     private string $baseUri = 'https://api.example.com';
12
13     public function __construct() {
14         $callback = \Closure::fromCallable([$this, 'handleRequest']);
15         parent::__construct($callback, $this->baseUri);
16     }
17
18     private function handleRequests(string $method, string $url): MockResponse
19     {
20         if ($method === 'GET' && $url == $this->baseUri.'/hello') {
21             return new MockResponse(
22                 json_encode(['bestConference' => 'API-Platform Con 2022'], JSON_THROW_ON_ERROR),
23                 ['http_code' => Response::HTTP_OK]
24             );
25         }
26
27         throw new \UnexpectedValueException("Mock not implemented: $method/$url");
28     }
29 }
```

# TESTER L'INTESTABLE

## TESTER UNE API DISTANTE ... SANS L'APPELER

```
use Elastic\Elasticsearch\ClientBuilder;
use Elastic\Elasticsearch\Response\Elasticsearch;
use Http\Mock\Client;
use Nyholm\Psr7\Response;

$mock = new Client(); // This is the mock client

$client = ClientBuilder::create()
    ->setHttpClient($mock)
    ->build();

// This is a PSR-7 response
$response = new Response(
    200,
    [Elasticsearch::HEADER_CHECK => Elasticsearch::PRODUCT_NAME],
    'This is the body!'
);
$mock->addResponse($response);

$result = $client->info(); // Just calling an Elasticsearch endpoint

echo $result->asString(); // This is the body!
```



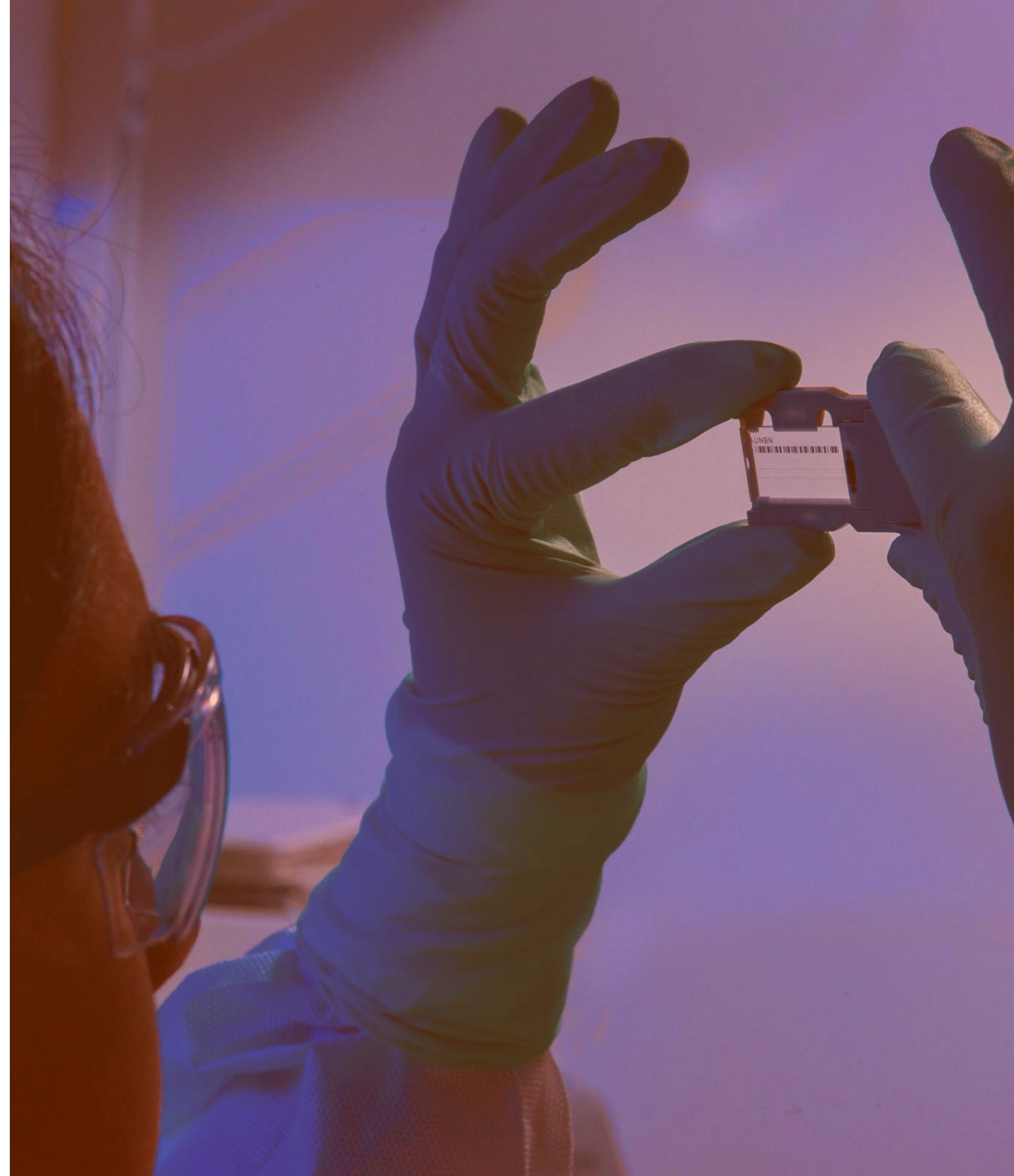
# TESTER LA SPEC OPENAPI

Avec le package NPM `@apidevtools/swagger-cli`

Utile en cas de surcharge complexe :

```
App\OpenApi\JwtDecorator:  
  decorates: 'api_platform.openapi.factory'  
  autoconfigure: false
```

```
php bin/console api:openapi:export -o openapi.json  
./node_modules/.bin/swagger-cli validate openapi.json
```





# TESTER LES ACL

Si votre API est hautement authentifié, mettez en place dès le début une vérification des ACL poussée sur chaque endpoint.  
En cas de régression vous direz merci 😊

- Générer un client par rôle
- Tester les retours avec :
  - assertMatchesResourceCollectionJsonSchema
  - assertMatchesResourceItemJsonSchema

```
$response = $this->getAdminClient()->request('GET', '/arya/organizations');  
self::assertResponseIsSuccessful();  
self::assertCount(1, $response->toArray()['hydra:member']);  
self::assertEquals(1, $response->toArray()['hydra:totalItems']);  
self::assertMatchesResourceCollectionJsonSchema(Organization::class);
```

# TOUR D'HORIZON **DES OUTILS**

# LES OUTILS D'ANALYSE DE CODE (1/3)

## PHPCS (PHP CODESNIFFER)

Teste si le formatage de vos fichiers suit bien un standard (PSR, Zend ...)  
Permet l'autocorrection sur une majorité d'erreurs.

```
composer require --dev "squizlabs/php_codesniffer=*" 
```



# LES OUTILS D'ANALYSE DE CODE (2/3)

## PHPCPD

### Détecteur de copier/coller

# Exemple sur le code api-platform v3  
phpcpd 6.0.3 by Sebastian Bergmann.

Found 171 clones with 5920 duplicated lines in 143 files:

- /home/odoucet/projects/core/tests/Doctrine/Odm/Filter/RangeFilterTest.php:31-337 (306 lines)  
/home/odoucet/projects/core/tests/Doctrine/Orm/Filter/RangeFilterTest.php:28-334
- /home/odoucet/projects/core/tests/Doctrine/Odm/Filter/OrderFilterTest.php:34-182 (148 lines)  
/home/odoucet/projects/core/tests/Doctrine/Orm/Filter/OrderFilterTest.php:32-180

...

4.87% duplicated lines out of 121512 total lines of code.

Average size of duplication is 34 lines, largest clone has 306 of lines

```
composer require --dev sebastian/phpcpd
```

# LES OUTILS D'ANALYSE DE CODE (3/3)

## PHPSTAN

*Alternatives : psalm ; phan ; exakat*

- Plusieurs niveaux de tolérance
- Plugin VSCode pour colorer les lignes qui posent problème
- Gère du typage de variable avancée (structures complexes de tableaux, ...)
- Extensions Doctrine, PHPUnit, Symfony ...

```
composer require --dev ekino/phpstan-banned-code  
phpstan/extension-installer phpstan/phpstan  
phpstan/phpstan-deprecation-rules phpstan/phpstan-  
doctrine phpstan/phpstan-symfony
```

# LES OUTILS D'ANALYSE DE CODE (3/3)

## PHPSTAN

```
/** @var array<int, \User> $users */  
$users = UserLoaders::loadUsers($user_ids);  
  
/** @param array{scheme:string,host:string,path:string} $parsed_url */  
function showUrl(array $parsed_url) { ... }
```



# LE COUTEAU SUISSE DU DEVELOPPEUR

## RECTOR

Aide à l'upgrade de versions de PHP :

- Constructor property promotion
- Annotation to attribute
- Switch to match
- `get_class()` vers `::class`

Transpilation de code PHP


```
composer require --dev rector/rector
```

# LE PETIT NOUVEAU A SURVEILLER

## INFECTION

Teste des variations de code et vérifie que vos tests unitaires les trouvent.

```
public function hasErrors(): bool
{
    return count($this->errors) > 0;
}
```



```
public function hasErrors(): bool
{
-   return count($this->errors) > 0;
+   return count($this->errors) >= 0;
}
```

```
public function hasErrors(): bool
{
-   return count($this->errors) > 0;
+   return count($this->errors) < 0;
}
```

```
public function hasErrors(): bool
{
-   return count($this->errors) > 0;
+   return count($this->errors) > 1;
}
```

### Metrics:

Mutation Score Indicator (MSI): 47%

Mutation Code Coverage: 67%

Covered Code MSI: 70%

# TotalDefeatedMutants = KilledCount + TimedOutCount + ErrorCount;

# MSI = (TotalDefeatedMutants / TotalMutantsCount) \* 100;







# UNE CI/CD **EFFICACE & LISIBLE**

# UNE CI/CD EFFICACE

- Rapide
- Maintenance faible ou nulle
- Fait gagner du temps sur des tâches récurrentes et rébarbatives



# UNE CI/CD RAPIDE

[GITLAB] Paralléliser l'exécution des tâches avec les *stages* et *needs/dependencies*.

```
image: alpine
```

```
stages:
```

- compile
- test
- package

```
compile-dev:
```

```
  stage: compile  
  script: xxx
```

```
compile-prod:
```

```
  stage: compile  
  script: xxx
```

```
test:
```

```
  stage: test
```

```
  needs:
```

- compile-dev
- ```
  script: xxx
```

```
pack-gz:
```

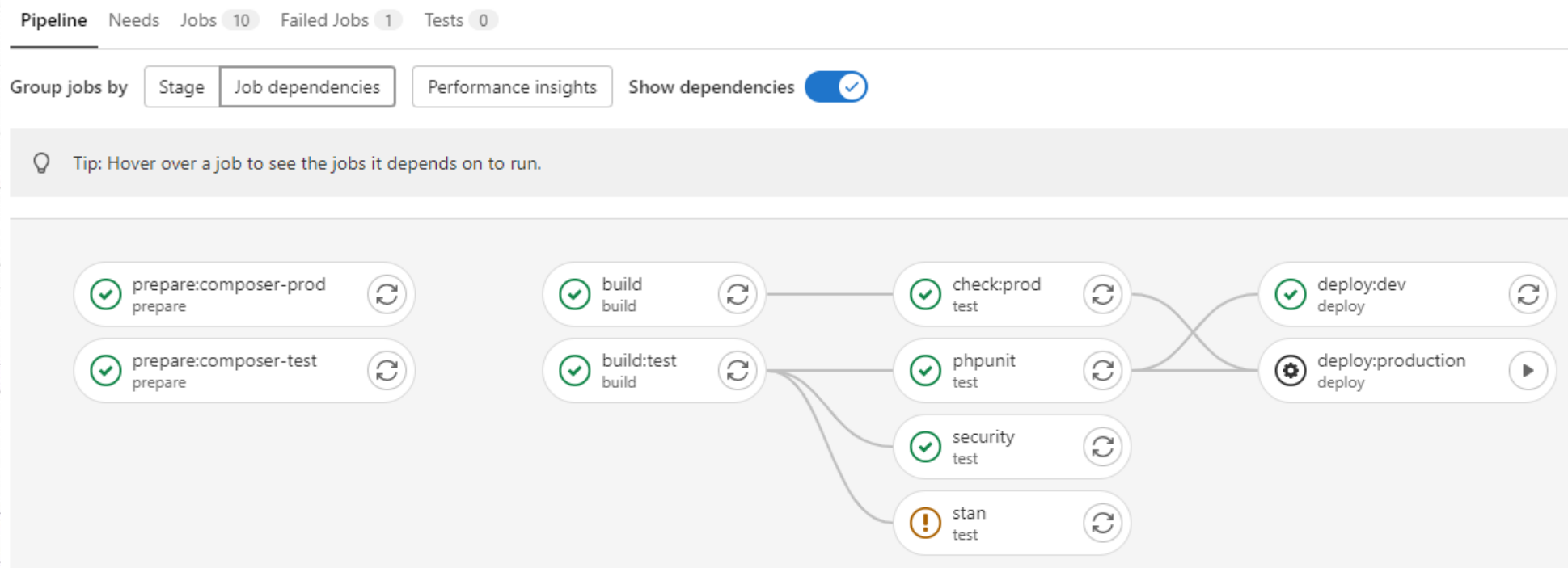
```
  stage: package  
  script: xxx  
  needs: ["test"]
```





# UNE CI/CD RAPIDE

[GITLAB] Paralléliser l'exécution des tâches avec les *stages* et les *dependencies*.



# UNE CI/CD RAPIDE

## BUILDKIT

*Alternatives : buildah ; Kaniko*

- Gère très bien le multi-stage
- Beaucoup plus rapide
- Pas besoin d'exécuter le container en mode privilégié (fonctionne sans root)
- Images plus légères avec moins de couches
- Utilisation du cache beaucoup plus simple

```
image : docker.io/moby/buildkit:rootless
variables:
  BUILDKITD_FLAGS: --oci-worker-no-process-
  sandbox
buildctl-daemonless.sh build \
  --frontend=dockerfile.v0 \
  --local context=. \
  --local dockerfile=. \
  --opt target=prod \
  --import-cache type=registry,ref=${IMAGE} \
  --export-cache type=inline \
  --output type=image,name=${IMAGE},push=true
```

# UNE CI/CD RAPIDE

[ALL] Remplacer phpunit par paratest

Exécution en parallèle de vos tests phpunit  
8 cœurs ? 8 fois plus rapide!

```
composer require --dev brianium/paratest
```





# UNE CI/CD LISIBLE

- Un pipeline d'exécution propre, avec des dépendances claires
- Une remontée d'information maximales de tous les outils utilisés





# UNE CI/CD LISIBLE

Sur Gitlab :

[https://docs.gitlab.com/ee/ci/yaml/artifacts\\_reports.html](https://docs.gitlab.com/ee/ci/yaml/artifacts_reports.html)

- \_ Rapports d'accessibilité (FRONTEND)
- \_ Rapports de performance **PREMIUM**
- \_ Couverture de code
- \_ Qualité de code

```
8 + messages,
9 + selectorGroups,
10 + addSelectors
11
12
13
14
15 + if (
16 +   rule &&
17 +   rule.parent &&
18 +   rule.parent.type !== 'atrue' &&
19 +   !(\r\n|\r|\n)/g, ' ');
```

New code quality degradations on this line

Critical - Consider simplifying this complex logical expression.

```
JS lib/math/add.js
1 + module.exports = (a, b) => {
2 +   console.log(a, b);
3 +   return a + b;
4 + };
5
6 + const deadCode = (a, b) => {
7 +   return a - b;
8 + };
```

No Test Coverage

Test summary contained 3 failed out of 3 total tests

rspec found 1 failed out of 1 total test

Failed 1 time in master in the last 14 days User#full\_name returns first\_name + last\_name

jest found 2 failed out of 2 total tests

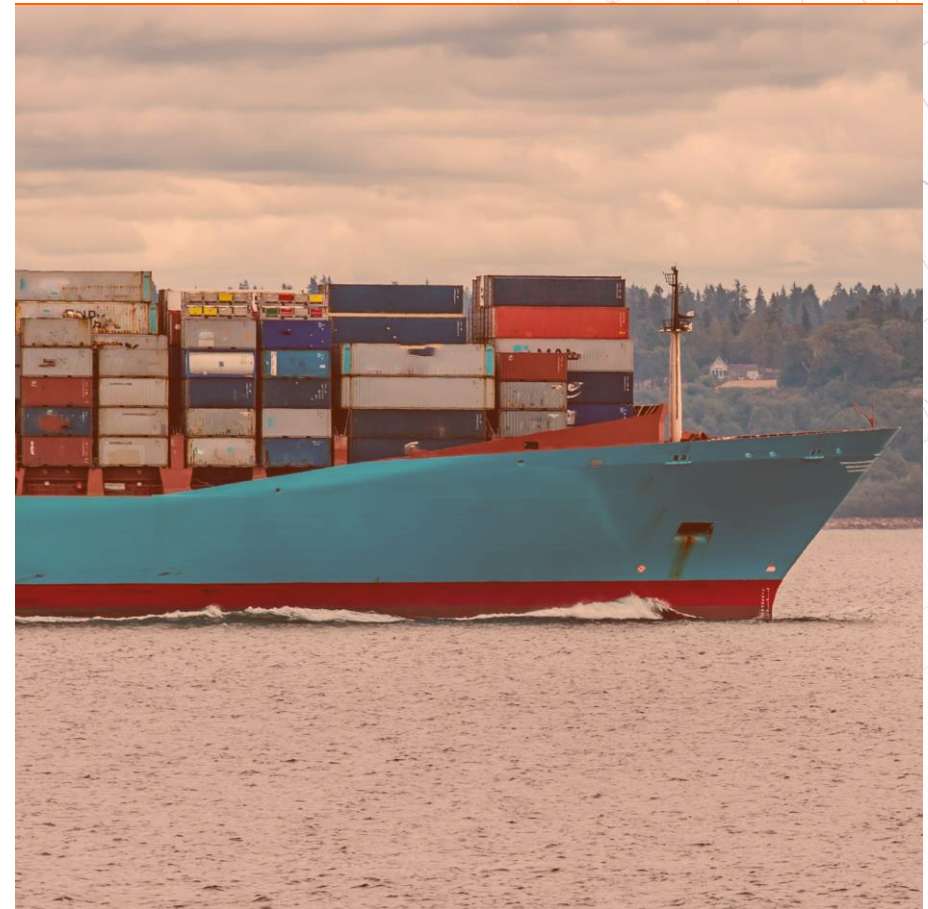
New Calculator #add returns the sum of the 2 given numbers

Failed 1 time in master in the last 14 days Calculator #subtract returns the difference between the 2 given numbers

# ET EN LOCAL ?

- Un fichier docker-compose.yml qui build les images en local avec le même Dockerfile
- Quelques raccourcis pour exécuter les commandes dans les docker (via un Makefile versionné dans le projet)

```
make load-fixtures  
make test filter=testFoo  
make get-token  
make composer-update
```



# PRÊT POUR LE 100% DE COUVERTURE ?



# MERCI !

N'hésitez pas à me solliciter aujourd'hui !



— Les sources des exemples :  
<https://github.com/odoucet/apiplatformcon2022>

— Pour poursuivre les discussions

**Olivier DOUCET**

[odoucet@oxeva.fr](mailto:odoucet@oxeva.fr)

Twitter [@ezameku](https://twitter.com/ezameku)