

Audit de performance
-- Blackfire --
Environnement
production

Configuration serveur

- Solution : Wampserver 3.1.3
- Apache : 2.4.23
- PHP : 7.1.8
- MySQL : 5.7.14
- Symfony 3.1 (environnement production)

Blackfire

Analyse du comportement du code basée à l'aide des métriques.

```
# .blackfire.yml
tests:
  "All pages are fast":
    path: "/*"
    assertions:
      - main.wall_time < 50ms
      - main.memory < 2Mb

  "[SQL] No more than 8 queries SQL":
    assertions:
      - metrics.sql.queries.count <= 8

  "[SQL] limit the number of DB connections":
    assertions:
      - metrics.sql.connections.count <= 1

...
```

Les métriques peuvent nous mettre en garde sur des mauvaises pratiques.

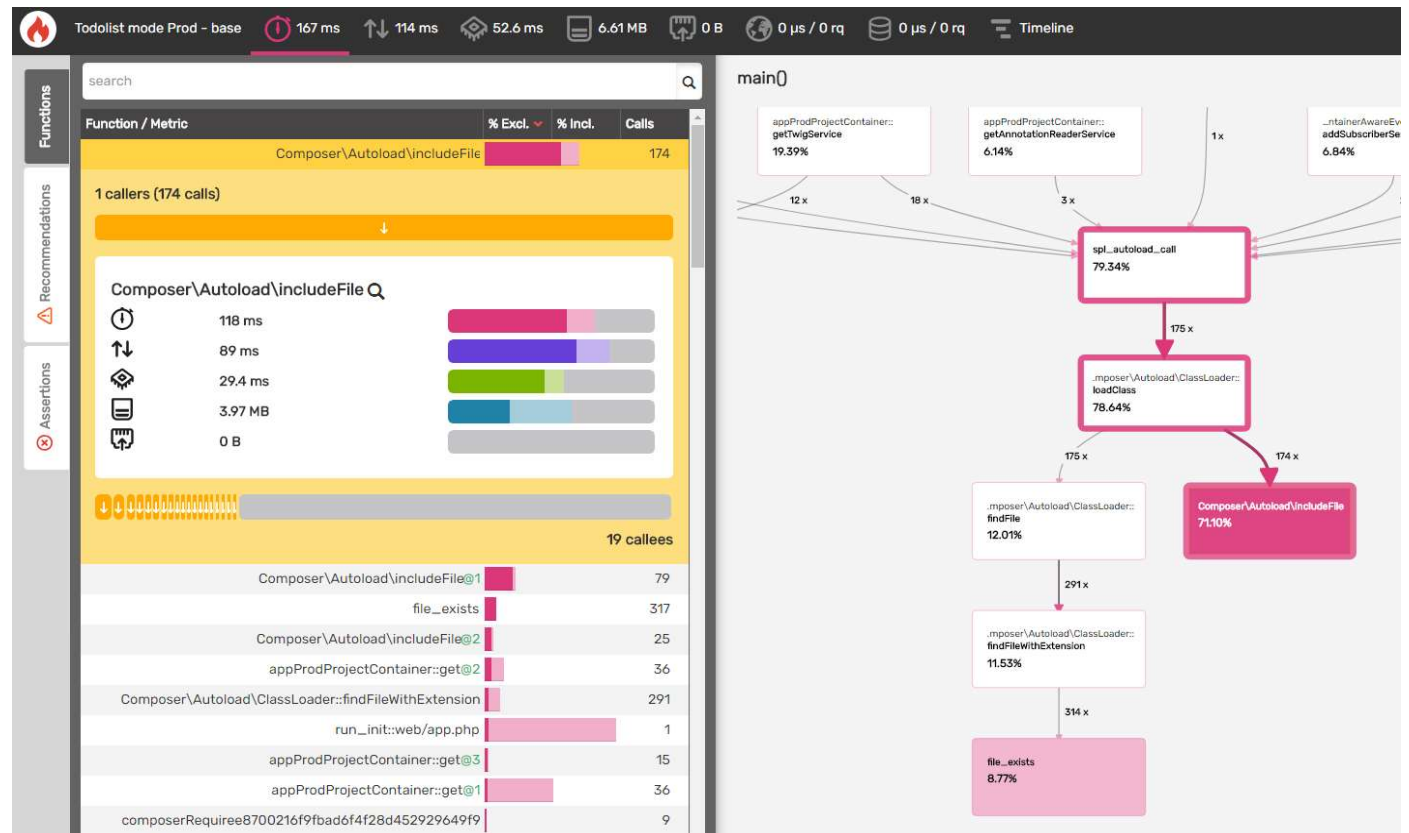
Le fichier de configuration /blackfire.yml répond à ce besoin.

<https://blackfire.io/docs/reference-guide/metrics>

✗ All pages are fast	main.wall_time 449ms < 50ms
	main.memory 12.1MB < 2Mb
✗ [SQL] No more than 8 queries SQL	metrics.sql.queries.count 21 <= 8
✓ [SQL] limit the number of DB connections	metrics.sql.connections.count 0 <= 1
✓ [TWIG] Twig displays and renders	metrics.twig.display.count 0 + metrics.twig.render.count 0 < 5
✓ [SF] Symfony events dispatched	metrics.symfony.events.count 0 < 10
✗ [Good practice] Symfony Http cache activated	metrics.symfony.http_cache.count 0 == 1
✓ [Good practice] never send emails synchronously	metrics.emails.sent.count 0 == 0
✓ [Good practice] no Twig/Smarty compilation	is_dev() false or metrics.twig.compile.count 0 == 0
	is_dev() false or metrics.smarty.compile.count 0 == 0
✓ [Good practice] no Symfony metadata checks	is_dev() false or metrics.symfony.config_check.count 0 == 0
✗ [Good practice] no Doctrine parsing	is_dev() false or (metrics.doctrine.annotations.parsed.count 0 + metrics.doctrine.annotations.read.count 0 + metrics.doctrine.dql.parsed.count 0 + metrics.doctrine.entities.metadata.count 2 + metrics.doctrine.proxies.generated.count 0) == 0
✓ [Good practice] no YAML loaded	is_dev() false or metrics.symfony.yaml.reads.count 0 == 0
✓ [Good practice] Assetic controller must not be called (assets should be dumped)	is_dev() false or metrics.assetic.controller.calls.count 0 == 0

/login

Profil de référence



1^{er} Objectif : Eradiquer les 317 appels à la fonction PHP file_exists

Composer

Agir sur la configuration

➤ Optimisation : Génération d'une carte de classe

- "optimize-autoloader": true

```
"autoload": {  
    "psr-4": {  
        "": "src/"  
    },  
    "optimize-autoloader": true,  
    "classmap": [  
        "app/AppKernel.php",  
        "app/AppCache.php"  
    ]  
},
```

- \$ composer dump-autoload -o - -no-dev

- En réalisant une carte des classes, Composer pour une classe retourne instantanément le chemin sans aucune vérification de l'existence du fichier. D'où la disparition des 317 appels à la fonction PHP `file_exists` visible sur le prochain rapport d'analyse.
- En contrepartie, la classe `/vendor/composer/autoload_static.php` est sollicité.

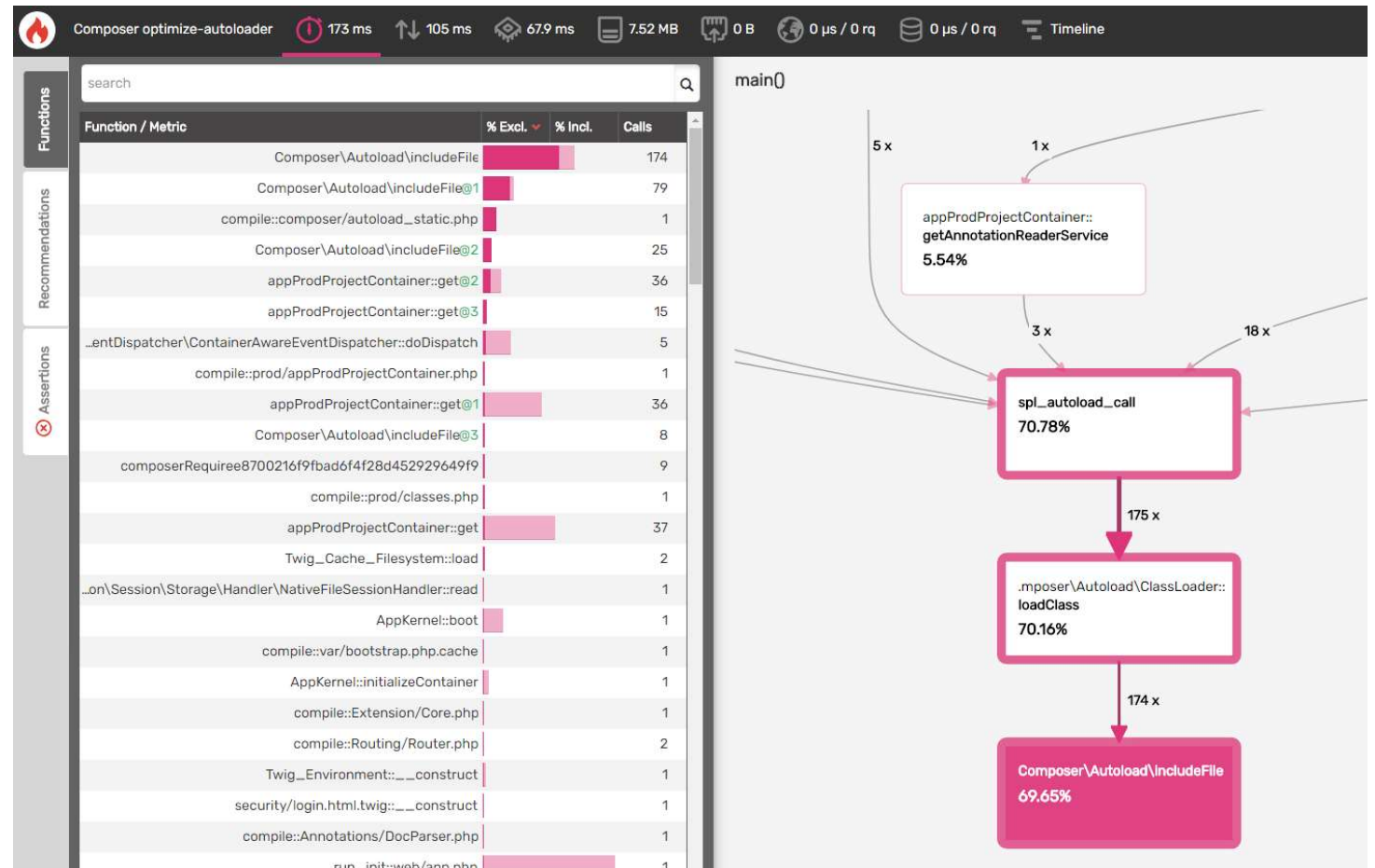
<https://getcomposer.org/doc/articles/autoloader-optimization.md>

/login

Profil de comparaison

Avec
l'autochargeur
optimisé

Plus d'appel à
file_exists

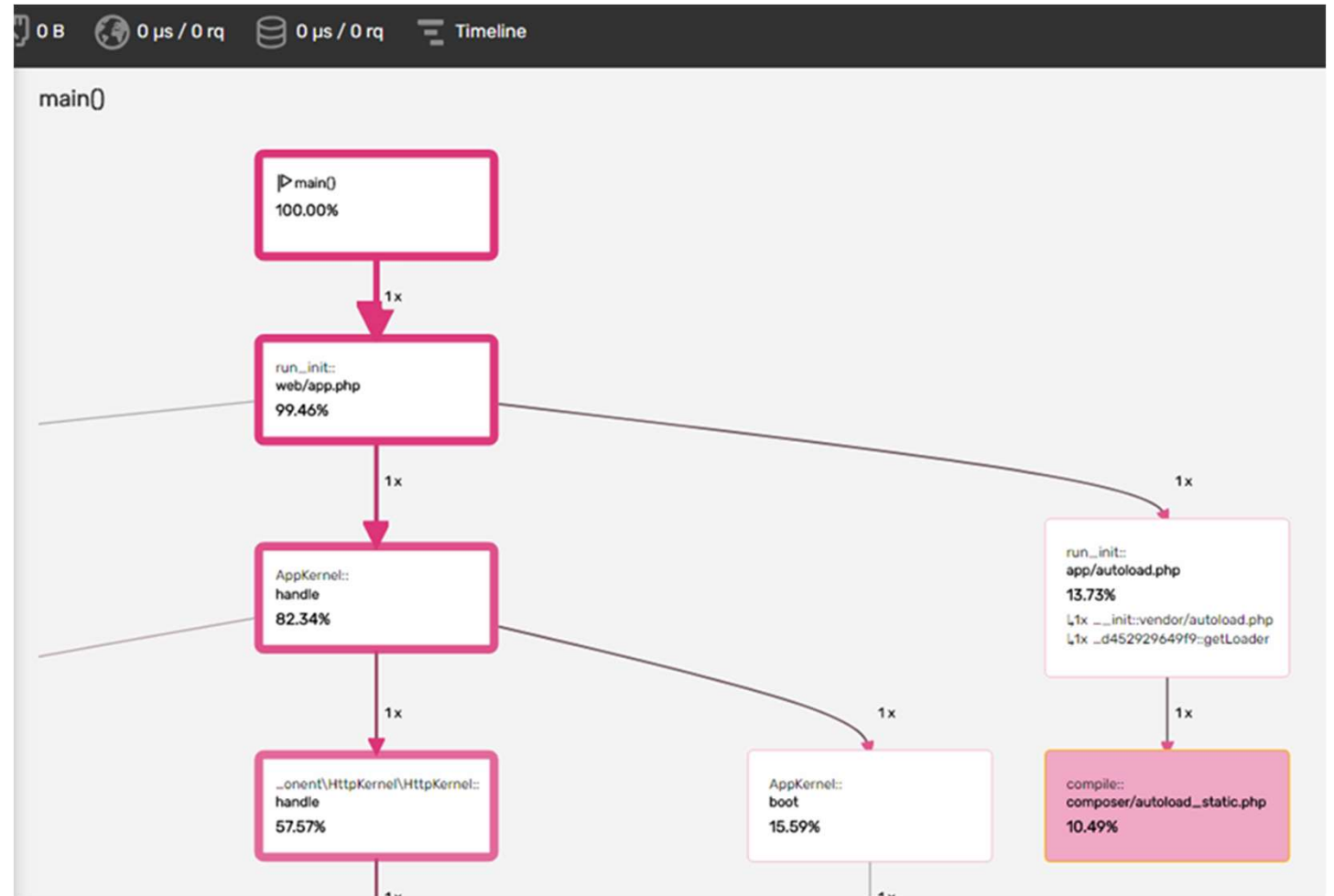


/login

Profil de comparaison

Avec
l'autochargeur
optimisé

La charge s'est
reportée sur
autoload_static



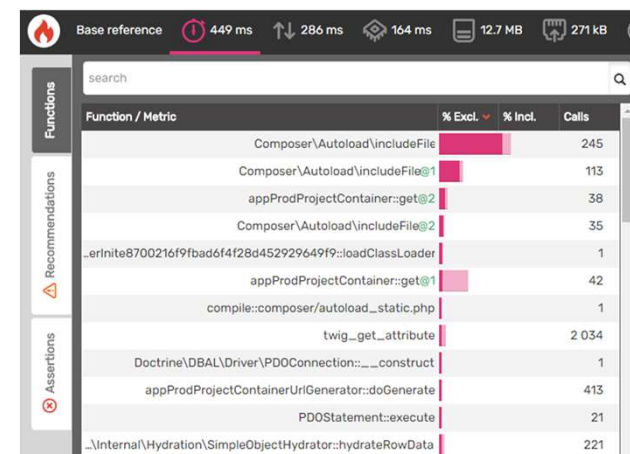
/tasks

Profil de référence

Profilage des requêtes HTTP à partir de l'interface CLI et usage de CURL

```
blackfire --samples=1 curl "http://oc_todolist.local/app.php/tasks" -H "Connection: keep-alive" -H "Cache-Control: max-age=0" -H "Upgrade-Insecure-Requests: 1" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.87 Safari/537.36" -H "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8" -H "Accept-Encoding: gzip, deflate" -H "Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7,es;q=0.6" -H "Cookie: __blackfire=NO_CACHEo.9148393663371772; PHPSESSID=lonv1tmav95otiiad51hd2kbev"
```

Résultat du profilage



Profilage de référence sur l'URL /tasks pour une comparaison ultérieure avec l'auto-chargeur de classes optimisé et activation de OPcache.

Temps d'exécution 449ms

/tasks

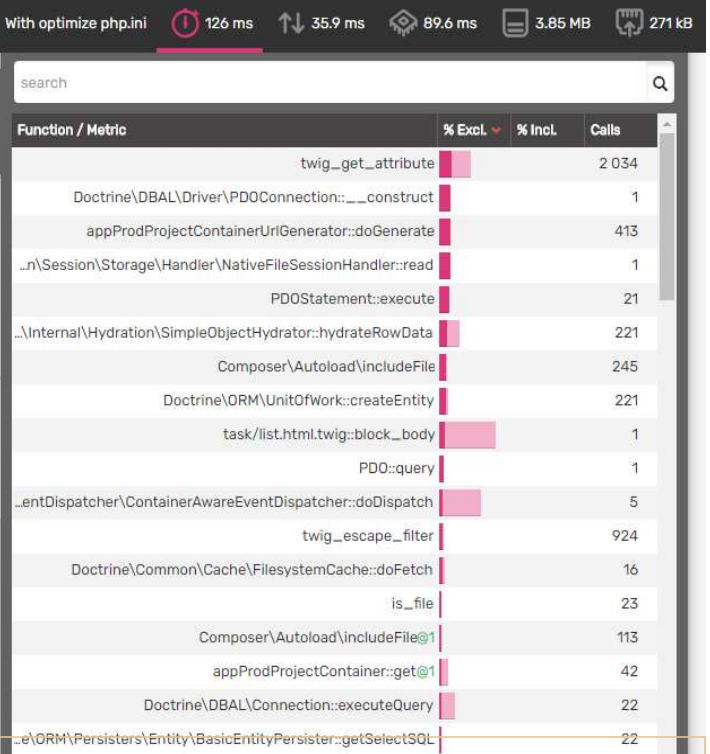
Profil de
comparaison

- Autochargeur de classes optimisé
- Activation de OPcache
- Ajustement des paramètres **php.ini** selon la documentation de Symfony.

<https://symfony.com/doc/current/performance.html>

Gain de performance considérable
De 449 ms à 126 ms en temps d'exécution

❖ Toutefois attention à bien comprendre le fonctionnement de OPcache.

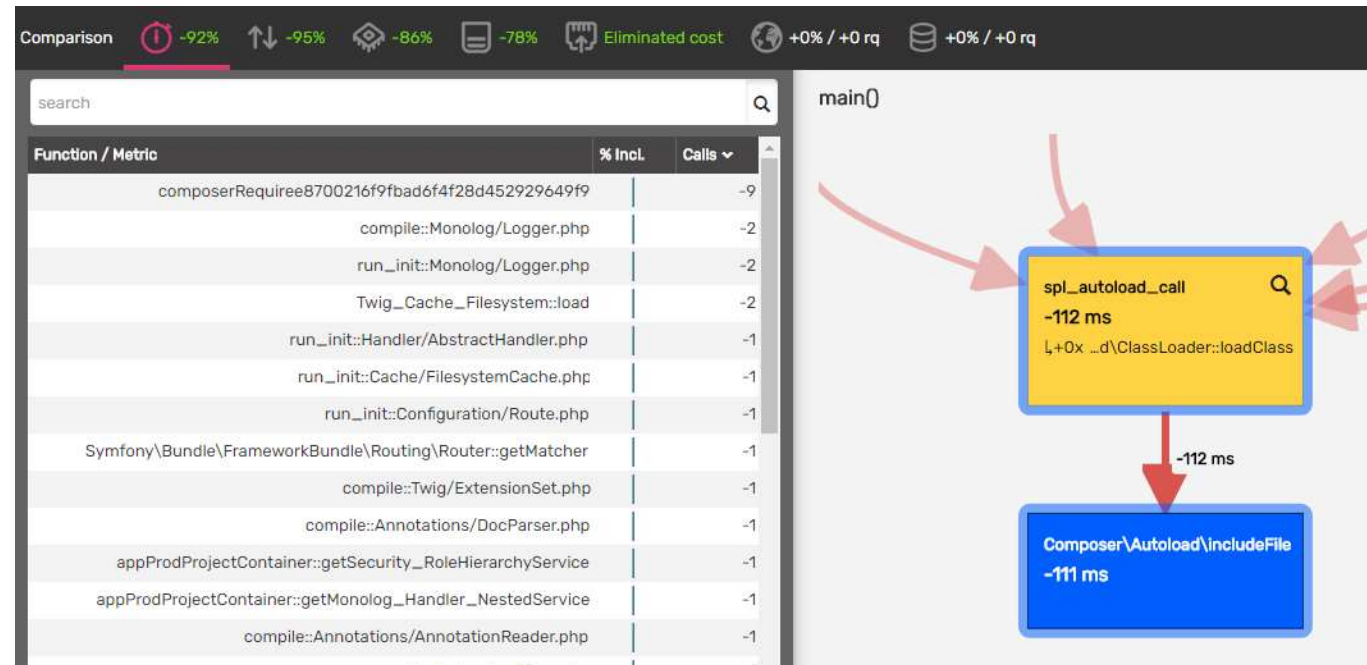


Function / Metric	% Excl.	% Incl.	Calls
twig_get_attribute			2 034
Doctrine\DBAL\Driver\PDOConnection::__construct			1
appProdProjectContainerUrlGenerator::doGenerate			413
...n\Session\Storage\Handler\NativeFileSessionHandler::read			1
PDOStatement::execute			21
...Internal\Hydration\SimpleObjectHydrator::hydrateRowData			221
Composer\Autoload\includeFile			245
Doctrine\ORM\UnitOfWork::createEntity			221
task/list.html.twig::block_body			1
PDO::query			1
...entDispatcher\ContainerAwareEventDispatcher::doDispatch			5
twig_escape_filter			924
Doctrine\Common\Cache\FilesystemCache::doFetch			16
is_file			23
Composer\Autoload\includeFile@1			113
appProdProjectContainer::get@1			42
Doctrine\DBAL\Connection::executeQuery			22
...e\ORM\Persisters\Entity\BasicEntityPersister::getSelectSQL			22

/Login

- Blackfire -

Outil de comparaison
de profilage



- L'outil de comparaison de profilage montre les écarts de gain ou perte de performance.
- Ici une comparaison automatique sur l'URL /login entre le profilage de référence et un profilage optimisé.