



Audit de Performance
et de qualité du code

Sommaire

1 – Performances avant modification	3
1.1 - Page d'accueil	3
1.2 - Création d'une tâche : affichage du formulaire.....	4
1.3 - Création d'un utilisateur : soumission du formulaire.....	5
2 – Performances après modification et optimisation.....	7
2.1 – Optimisation.....	7
2.1.1 - Utilisation d'OPcache	7
2.1.2 - Configuration du « realpath PHP Cache ».....	7
2.1.3 - Optimisation de l'autoloader de composer	7
2.2 – Comparaison des performances	8
3 – Qualité du code.....	9
3.1 – Outils utilisés	9
3.2 – Analyses sur différentes versions de l'application	9
3.3 – Analyse avant modifications	9
3.4 – Mise à jour de Symfony	10
3.5 – Analyse après modification et montée en version	10
3.6 – Analyse après correction des erreurs	11
3.7 – Qualité du rendu de l'application.....	12

1 – Performances avant modification

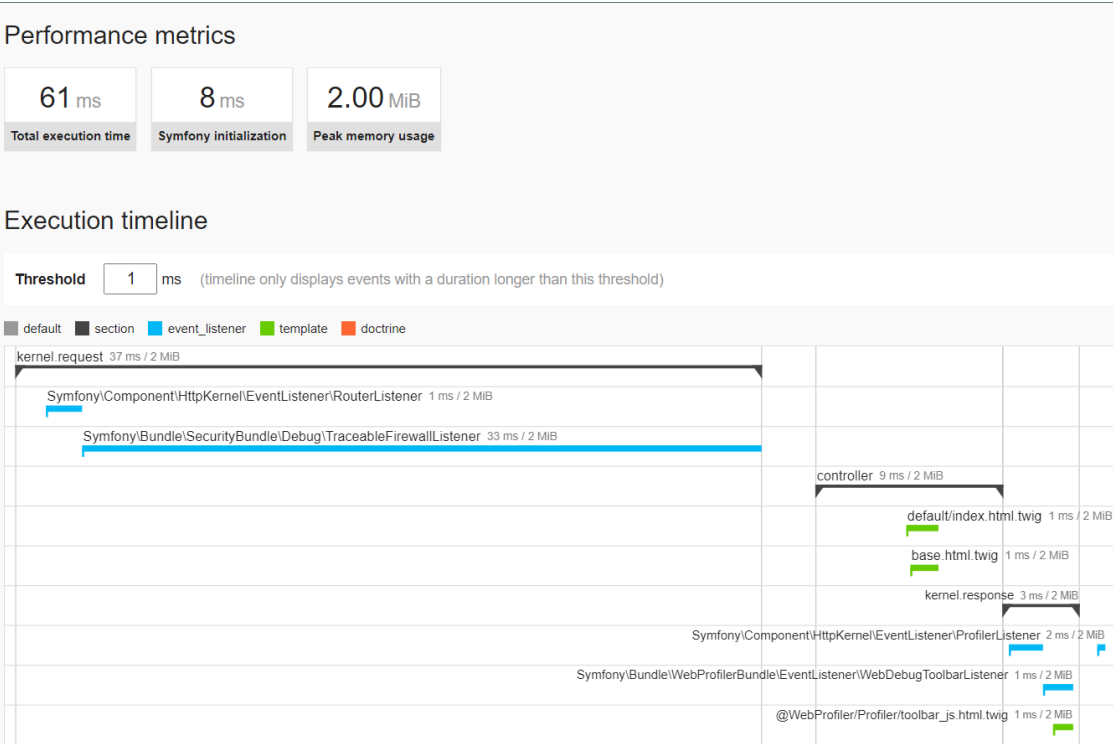
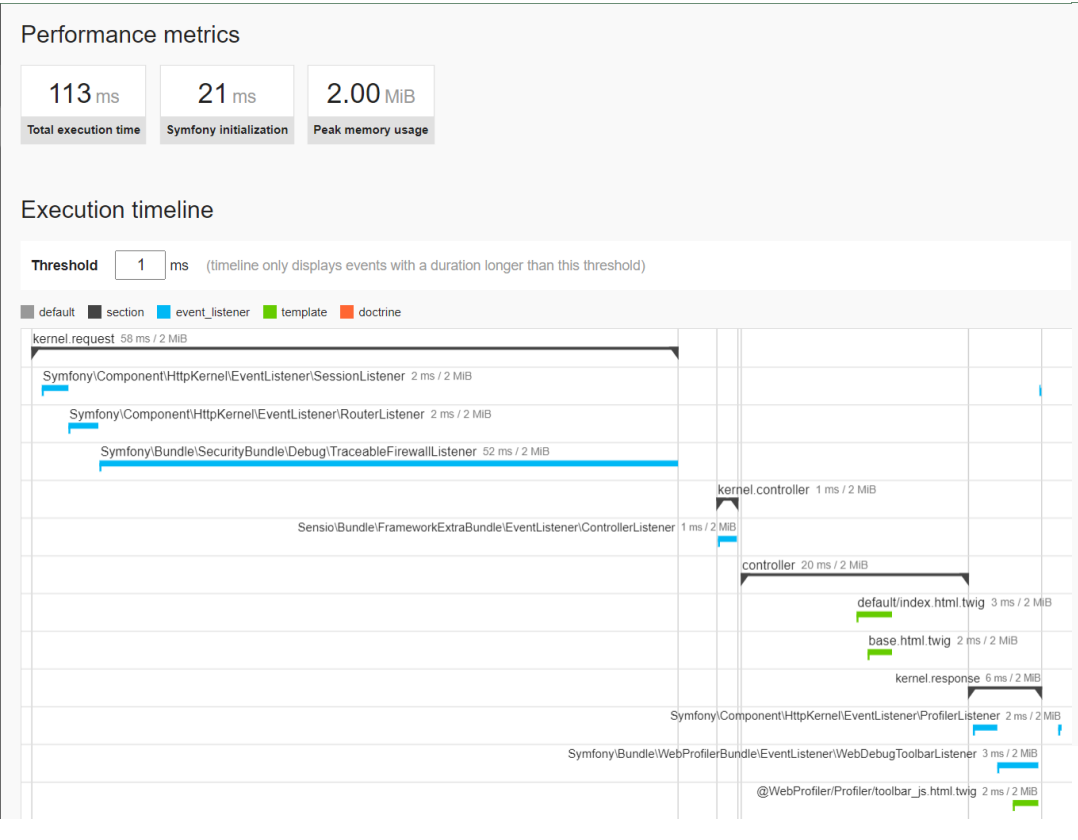
1.1 - Page d'accueil

Method: GET

```
/**
 * @Route("/", name="homepage")
 */
```

Sans cache

Avec cache



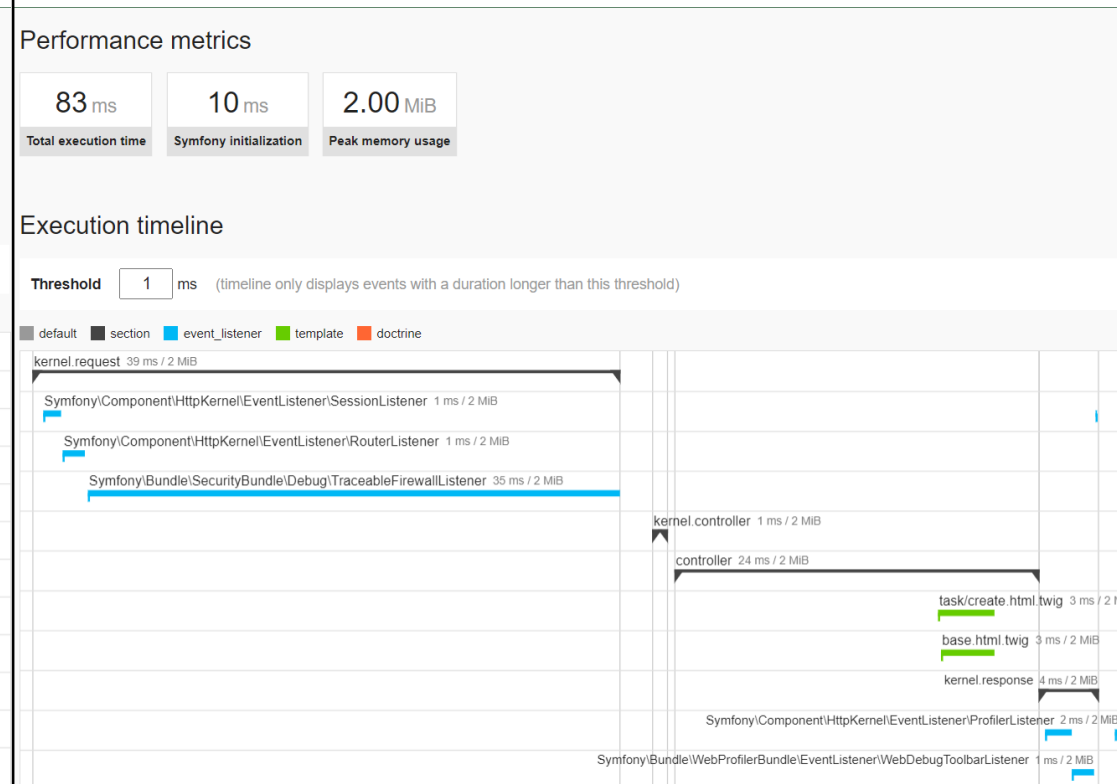
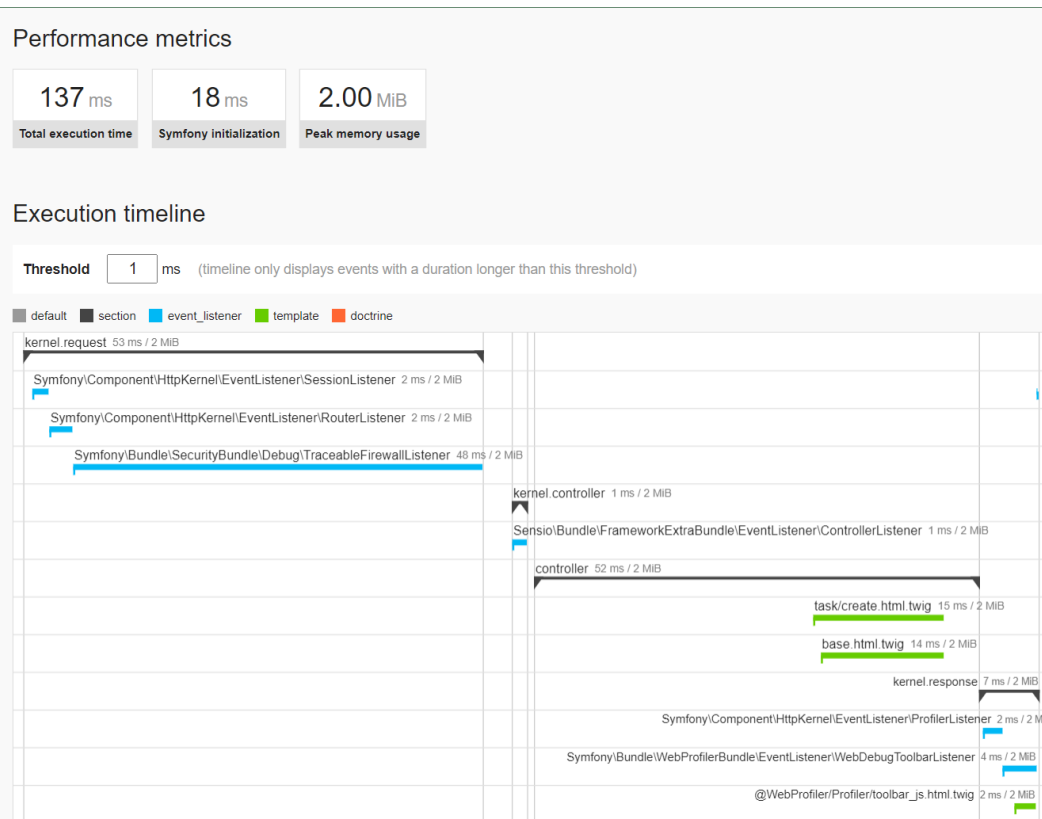
1.2 - Création d'une tâche : affichage du formulaire

Method: GET

```
/**
 * @Route("/tasks/create", name="task_create")
 */
```

Sans cache

Avec cache



Le composant TraceableFirewallListener qui gère la sécurité de l'application semble être long à charger comparer au RouterListener / SessionListener. Il met 48ms à charger alors que les 2 autres Listeners mettent seulement 2ms.

Toutes les autres pages ont également ce même problème de performance. Par exemple la page « homepage » a un temps de chargement de 113ms et 52ms sont utilisées par ce composant.

1.3 - Création d'un utilisateur : soumission du formulaire

Method: POST

```
/**  
 * @Route("/users/create", name="user_create")  
 */
```

Performance metrics

1152 ms	10 ms	2.00 MiB
Total execution time	Symfony initialization	Peak memory usage

Execution timeline

Threshold ms (timeline only displays events with a duration longer than this threshold)

■ default ■ section ■ event_listener ■ template ■ doctrine



On peut voir sur le profiler que le traitement du contrôleur prend 1099ms sur les 1152ms du temps de chargement. Ici on a divisé en 2 parties les lignes qui sont exécutées dans le contrôleur avec le composant stopwatch de symfony pour mieux identifier d'où vient le problème.

Ci-dessous les lignes de codes concernés :

```
/**
 * @Route("/users/create", name="user_create")
 */
no usages  ↳ Saro0h *
public function createAction(Request $request)
{
    $stopwatch = $this->get('debug.stopwatch');
    $stopwatch->start('création du formulaire');
    $user = new User();
    $form = $this->createForm( type: UserType::class, $user);
    $form->handleRequest($request);
    $stopwatch->stop('création du formulaire');

    if ($form->isValid()) {
        $stopwatch->start("création de l'utilisateur");
        $em = $this->getDoctrine()->getManager();
        $password = $this->get('security.password_encoder')->encodePassword($user, $user->getPassword());
        $user->setPassword($password);

        $em->persist($user);
        $em->flush();

        $this->addFlash( type: 'success', message: "L'utilisateur a bien été ajouté.");
        $stopwatch->stop("création de l'utilisateur");

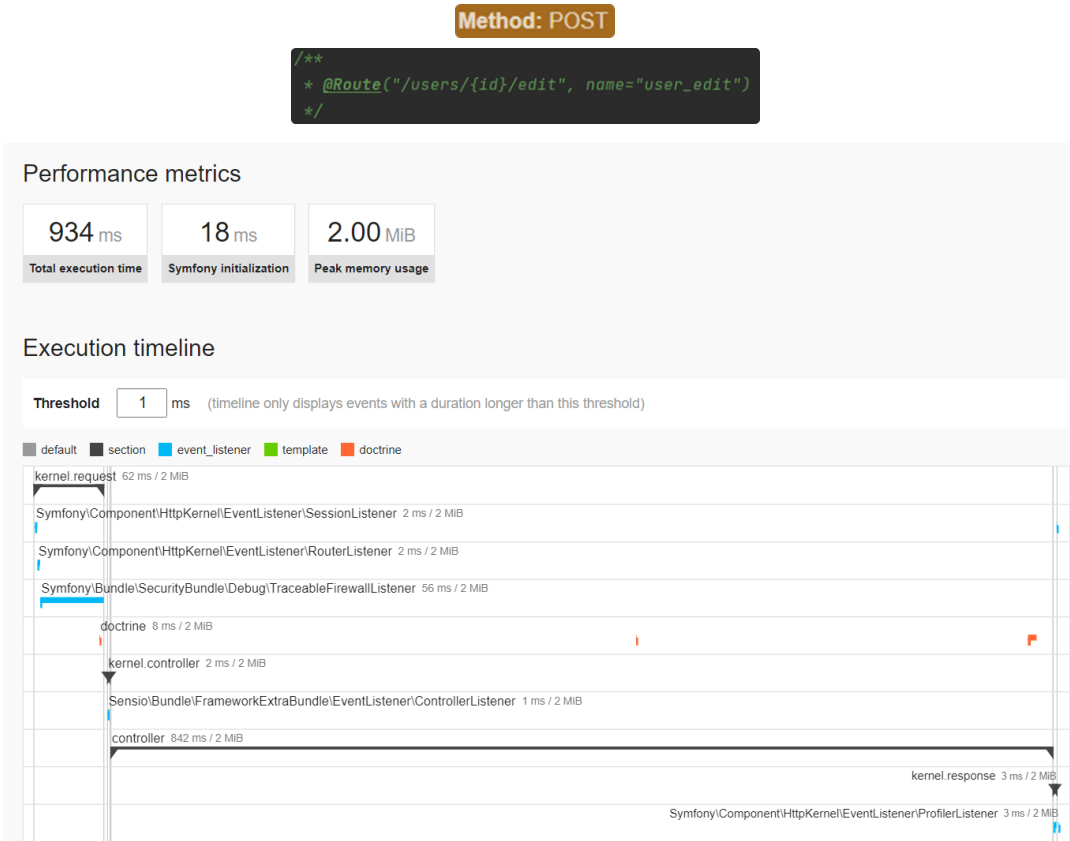
        return $this->redirectToRoute( route: 'user_list');

    }

    return $this->render( view: 'user/create.html.twig', ['form' => $form->createView()]);
}
```

Cela nous permet de voir que c'est la création du formulaire qui prend le plus de temps lors de l'exécution du contrôleur.

On retrouve également ce problème de performance sur la modification d'un utilisateur



2 – Performances après modification et optimisation

2.1 – Optimisation

Les performances de l'application ont été optimisées en se basant sur plusieurs recommandations faites par Symfony dans la documentation officielle :

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

2.1.1 - Utilisation d'OPcache

Fichier php.ini de PHP 8.2.0

```
; Determines if Zend OPcache is enabled
opcache.enable=1

; Determines if Zend OPcache is enabled for the CLI version of PHP
;opcache.enable_cli=0

; The OPcache shared memory storage size.
opcache.memory_consumption=256

; The amount of memory for interned strings in Mbytes.
;opcache.interned_strings_buffer=8

; The maximum number of keys (scripts) in the OPcache hash table.
; Only numbers between 200 and 1000000 are allowed.
opcache.max_accelerated_files=20000
```

2.1.2 - Configuration du « realpath PHP Cache »

Fichier php.ini de PHP 8.2.0

```
; Determines the size of the realpath cache to be used by PHP. This value should
; be increased on systems where PHP opens many files to reflect the quantity of
; the file operations performed.
; Note: if open_basedir is set, the cache is disabled
; https://php.net/realpath-cache-size
realpath_cache_size = 4096k

; Duration of time, in seconds for which to cache realpath information for a given
; file or directory. For systems with rarely changing files, consider increasing this
; value.
; https://php.net/realpath-cache-ttl
realpath_cache_ttl = 600
```

2.1.3 - Optimisation de l'autoloader de composer

Dans le fichier composer.json se trouvant à la racine du projet, la ligne suivante a été ajoutée :

```
"config": {
    "classmap-authoritative": true,
    "allow-plugins": {
        "composer/package-versions-deprecated": true,
        "symfony/flex": true,
        "symfony/runtime": true
    },
    "preferred-install": {
        "*": "dist"
    },
    "sort-packages": true
}
```

- Suppression du dossier vendor et du cache qui se trouve dans le dossier var/cache

- Lignes de commandes exécutées dans l'ordre suivant :

```
$ composer update -a --classmap-authoritative
```

```
$ composer dump-autoload -a --classmap-authoritative
```

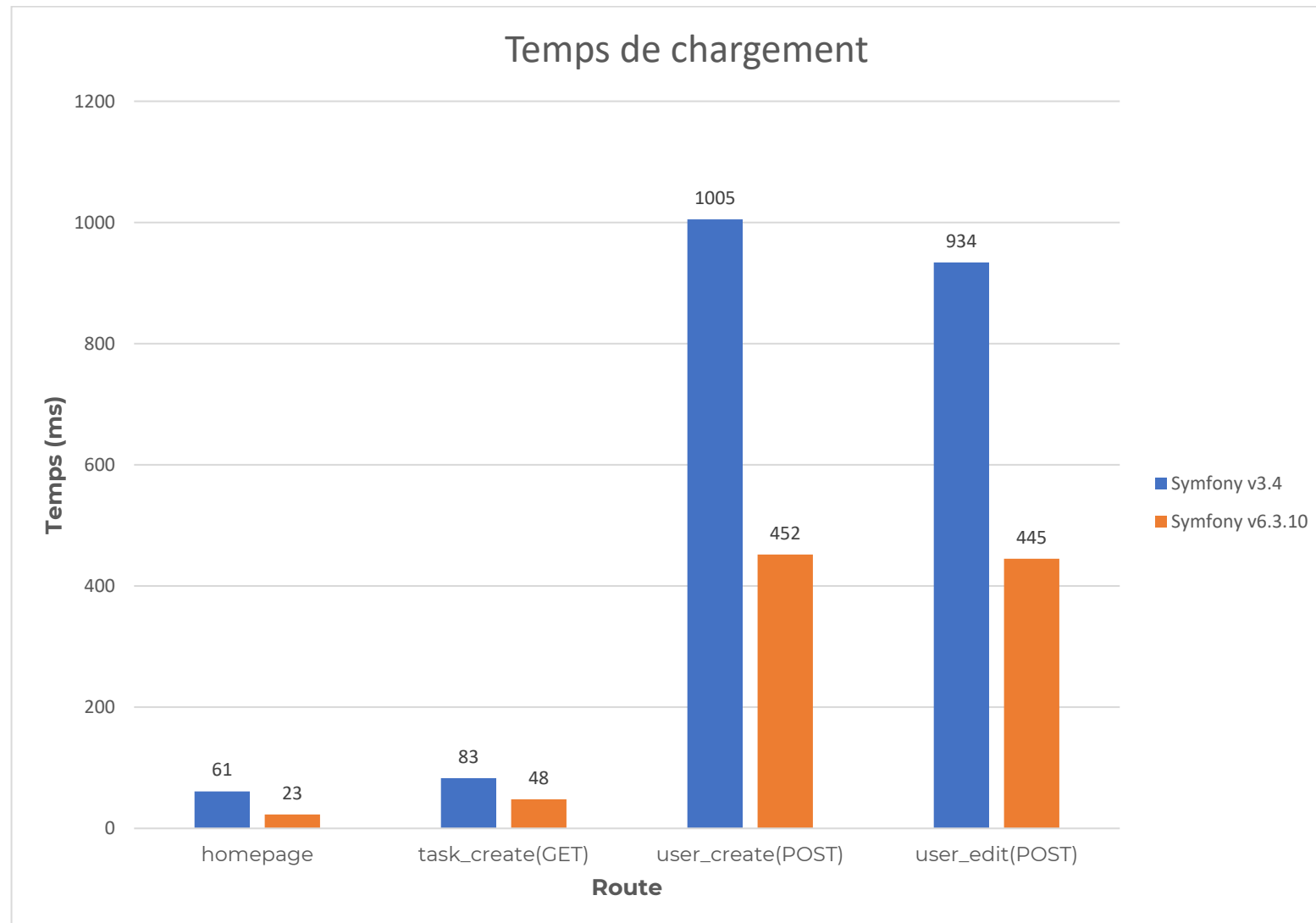
Cela permet à composer de générer une « carte des classes » lui permettant de retourner instantanément le chemin d'une classe si elle existe. Plus besoin d'utiliser le système de fichier, ce qui améliore grandement le temps chargement des classes.

Dès qu'une classe est rajoutée, cette manipulation doit être refaite.

Lien vers la documentation officielle de composer :

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

2.2 – Comparaison des performances



3 – Qualité du code

3.1 – Outils utilisés

L'analyse du code sera réalisée avec **Symfony Insight** qui est un outil spécialement conçu pour effectuer des analyses avancées sur des applications Symfony. Les erreurs trouvées dans l'analyse sont divisées en plusieurs catégories, allant des problèmes de performances aux exploits de sécurité potentiels.

3.2 – Analyses sur différentes versions de l'application

La version de Symfony utilisée par le projet étant assez ancienne (version 3.1 – sortie en 2016), une montée en version 3.4 a été réalisée pour faire une première analyse du code avant toute modification sur le projet

Les autres analyses ont été réalisées après la montée en version 6.3.8 de Symfony et la mise en place de toutes les fonctionnalités / corrections demandés.

3.3 – Analyse avant modifications

The screenshot shows the Symfony Insight analysis interface for the project 'TodoList_symfony3 #1'. The interface is divided into several sections:

- Header:** Shows the project name 'TodoList_symfony3 #1', the user 'nerym492', and the analysis date 'Analyzed 2 days ago by nerym492, duration: a minute'. There is an 'Analyze' button and an 'Edit configuration' link.
- Summary:** A bar indicates 'Changes: +52 suggestions' with a breakdown: 4 critical, 40 major, 6 minor, and 2 info.
- Filters:** Buttons for 'Suggestions (52)', 'Fixed', and 'Ignored'. A 'Stats' button is also present.
- Issues List:** A list of issues with details on severity and risk.
 - Issue 1: 'Your application failed to start'. Severity: Critical. Risk: Uninsured. Action: Read doc, Ignore all.
 - Issue 2: 'Your project must not rely on dependencies with known security issues'. Severity: Critical. Risk: Security. Action: Read doc, Ignore all.
 - Issue 3: 'Your project must not expose sensitive infrastructure configuration'. Severity: Critical. Risk: Data leak. Action: Read doc, Ignore all.
 - Issue 4: 'Your project must use a custom favicon instead of the default one'. Severity: Critical. Risk: Reputation. Action: Read doc, Ignore all.
 - Issue 5: 'Your project should use Doctrine migrations'. Severity: Major. Risk: Reliability. Action: Read doc, Ignore all.
 - Issue 6: 'Your project must not contain invalid instantiations'. Severity: Major. Risk: Reliability. Action: Read doc, Ignore all.
 - Issue 7: 'Your project should not use invalid parameter and return typehints'. Severity: Major. Risk: Reliability. Action: Read doc, Ignore all.
 - Issue 8: 'Your project should use return types'. Severity: Major. Risk: Reliability. Action: Read doc, Ignore all.
- Severity Legend:** 4 Critical, 40 Major, 6 Minor, 2 Info.
- Risk Legend:** 1 Data leak, 5 Productivity, 38 Reliability, 5 Reputation, 1 Security, 1 Uninsured.
- Developer Legend:** 47 Sorooh, 4 Collective, 1 Florian Pohu.
- Stats:** Lines of code: 2,847, Nb of suggestions: 52.
- Last commit:** A button to view the last commit.

Grâce à cette première analyse, on peut déjà voir qu'il y a des problèmes de sécurité ainsi que de potentielles fuites de données donnant des informations critiques sur la structure de l'application. Cette analyse n'a pas pu être réalisée en totalité car le démarrage de l'application a échoué lors de la simulation. On obtient alors la moins bonne note possible pour une analyse qui est de ne pas avoir de médaille (30/100).

Pour pouvoir faire une analyse complète, il faut donc mettre la version de Symfony à jour.

3.4 – Mise à jour de Symfony

La mise à jour a été faite en plusieurs étapes :

Symfony			
Ancienne version	Nouvelle version	PHP	Composer
3.1	3.4	7.4.33	2.2.0
3.4	4.0	8.2.0	2.6.5
4.0	4.4	8.2.0	2.6.5
4.4	5.0	8.2.0	2.6.5
5.0	5.4	8.2.0	2.6.5
6.0	6.3	8.2.0	2.6.5

Le code obsolète est mis à jour à chaque nouvelle version pour s'assurer que l'application fonctionne toujours correctement.

Les versions <4.0 utilisaient des versions de PHP et composer différentes.

Le changement le plus important a été la mise en place de Symfony flex qui est utilisé depuis la version 4.0. Cela a entraîné d'importants changements au niveau de la structure des dossiers et des fichiers de l'application.

3.5 – Analyse après modification et montée en version

nerym492 / TodoList #1

Analyzed 3 days ago by nerym492, duration: a minute Edit configuration Analyze

Changes: +42 suggestions 18 critical 4 major 17 minor 3 info

Suggestions (36) Fixed Ignored Stats

- Your project must not use PHP super globals 11 Read doc Ignore all Productivity Critical
- Your project must provide a favicon in its public directory Read doc Ignore all Reputation Critical
- Potential errors are silenced in your project Read doc Ignore all Reliability Major
- Your project should not contain "FIXME" comments Read doc Ignore all Productivity Major
- Your project should contain a robots.txt file Read doc Ignore all Reputation Major
- Your project session configuration could expose your users data Read doc Ignore all Legal data leak Major
- Your project should use the latest stable Symfony version Read doc Ignore all Productivity Minor

Severity

- 12 Critical
- 4 Major
- 17 Minor
- 3 Info

Risk

- 1 Legal data leak
- 19 Productivity
- 1 Reliability
- 15 Reputation

Developer

- 21 Florian Pohn
- 13 Collective
- 2 Saroth

Stats


Lines of code: 3,166

Nb of suggestions: 42

Faire la montée en version de Symfony a permis de corriger les problèmes de sécurité liés aux dépendances obsolètes et à la structure de l'application.


L'analyse est cette fois beaucoup plus complète que la précédente due au fait que l'application a pu démarrer correctement. On peut d'ailleurs voir que la note obtenue est passé de 30/100 à 25/100.

3.6 – Analyse après correction des erreurs

 nerym492 /
ToDoList #3


Analyzed 2 days ago, duration: a minute Edit configuration

Analyze ▼

<  >

Stats
Lines of code: **1,554**
Nb of suggestions: **0**

Last commit
[Change web manifest rule](#) by [Florian Pohu](#) 2 days ago.

 SymfonyInsight
Platinum medal

Changes: **-3 suggestions** 0 critical -1 major -2 minor 0 info

No suggestion Fixed (3) Ignored


Stats


Awesome!


Despite a very thorough analysis, SymfonyInsight couldn't find a single suggestion in the whole codebase of this project .

This is very rare, and is worthy of the Platinum medal.

Congratulations to all the developers of this project for such a high quality!

 Florian Pohu
102 commits

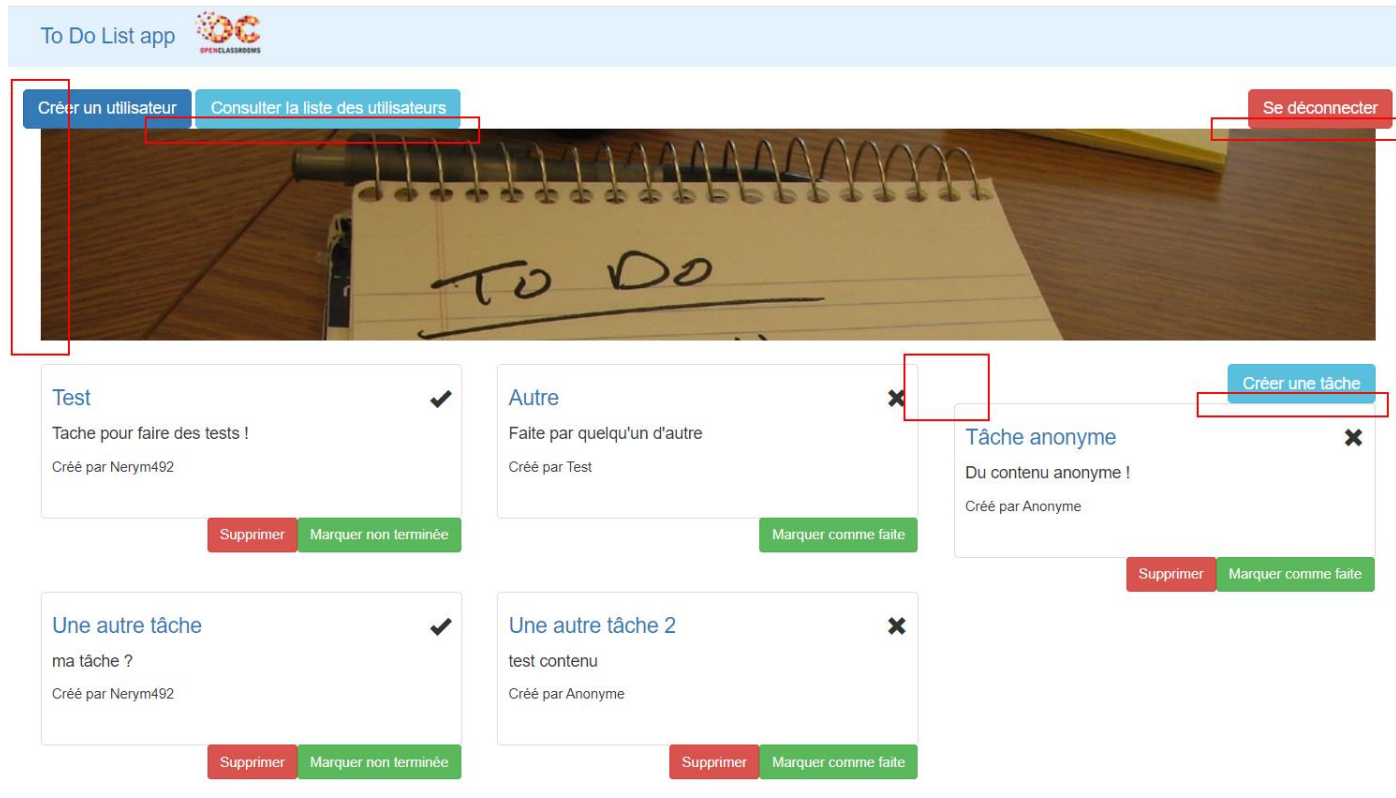
 Nerym492
12 commits

 Saro0h
2 commits

On obtient une médaille de platine pour cette analyse, c'est la meilleure note possible dans Symfony Insight.

3.7 – Qualité du rendu de l'application

```
/**  
 * @Route("/tasks", name="task_list")  
 */
```



Copyright © OpenClassrooms

Le rendu pourrait être amélioré sur certaines pages de l'application en rajoutant/modifiant des règles css ou en réorganisant les différentes vues.

Le template « base.html.twig » applique les mêmes règles de mise en forme sur toutes les pages ce qui pose des problèmes d'affichages dans certaines situations comme ci-dessus.