Audit de qualité

ToDo & Co est une startup dont le cœur de métier est une application de gestion des tâches quotidiennes.

L’entreprise vient tout juste d’être crée, et l’application a dû être développée rapidement pour permettre de présenter à de potentiels investisseurs que le concept est viable (Minimum Viable Product ou MVP) avec le Framework Symfony dans sa version 3.1.

La mission qui m’a été confier et de développer de nouvelles fonctionnalités, et d’analyser la qualité du code ainsi que ses performances.

Ce document explique les améliorations apportées au code et compare les performances avant et après modifications.

Sommaire

1 – Le code initial

* *1.1 Les premières optimisations : Upgrader la version de Symfony et PHP*
* *1.2 Mesure de performance avant et après l’upgrade (BlackFire)*

2 – Améliorations et nouvelles fonctionnalités

* *2.1 Coté Backend*
* *2.2 Coté Frontend*

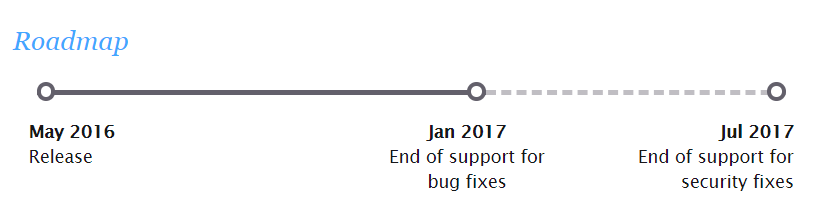
3 – Tests et performance du nouveau code

* Test unitaire et fonctionnel (PHP Unit)
* Test fonctionnel (Selenium)
* Mesure de performance finale (BlackFire)
* 1 – Le code initial

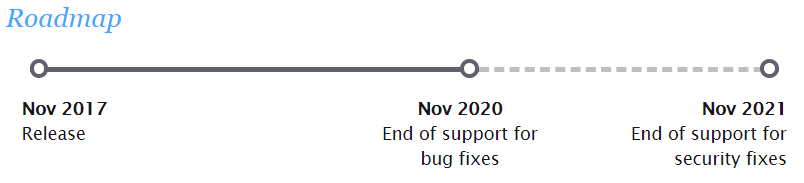
**1.1 Première optimisation : Upgrader la version de Symfony**

La version de Symfony initial de l’application est la 3.1.

Cette version à été mis en ligne en mai 2016 puis maintenu à jour jusqu’en janvier 2017 pour les bugs et jusqu’en juillet 2017 pour les correctifs de sécurité.



Passer sur la version 3.4 (*LTS = Long Term Support*) de Symfony permet de recevoir des mises à jour jusqu’en Novembre 2020 et des correctifs de sécurité jusqu’en Novembre 2021.



**Seconde optimisation : Upgrader la version de PHP**

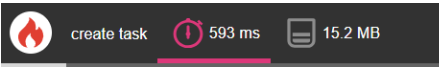
La version de PHP utilisé est la 5.5.9, passer à la version 7.2 de PHP augmentera les performance au niveau de l’utilisation des ressources et offrira la possibilité d’utilisé les dernières fonctionnalités de PHP comme le typage des variables qui rendra le code plus stable.

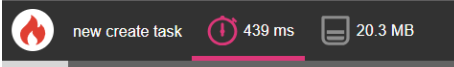
Comparaison des performances de Symfony 3.1 avec PHP 5 et la version 3.4 de Symfony avec PHP 7.2 d’après le site php benchmarks : <http://www.phpbenchmarks.com/fr/benchmark/apache-bench/php-7.2/select-version/symfony.html>



**1.2 Mesure de performance : Avant et après l’upgrade**

L’outil utilisé pour les mesures est [BlackFire.io](https://blackfire.io/) . Outil développé par Sensio Lab créateur du framework Symfony spécialisé dans les mesures d’applications PHP.





Mesure de la page Create Task avant et après l’upgrade (en environnement de développement)

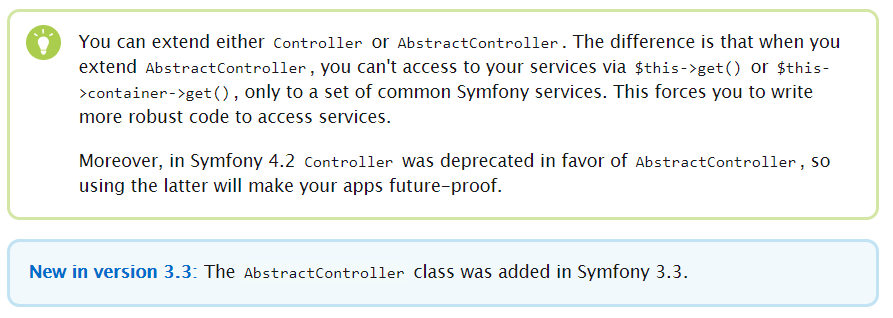
2 – Améliorations et nouvelles fonctionnalités

**2.1 Améliorations coté Backend**

Plusieurs modifications ont été apportées sur les contrôleurs.

* Première modification : Les contrôleurs étendaient de la classe **Controller** (classe déprécié sur les nouvelles versions de Symfony elle et remplacer par la classe **AbstractControler)** .

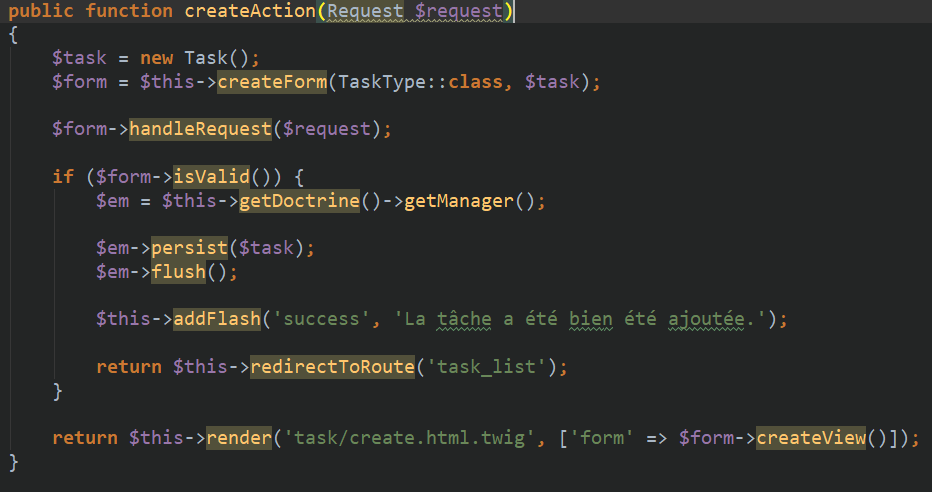
La classe Controller à donc été supprimer, mais pour de meilleur performance je n’ai pas étendu la classe vers sa remplaçante mais plutôt choisi d’injecter les services vraiment utiles aux contrôleurs.



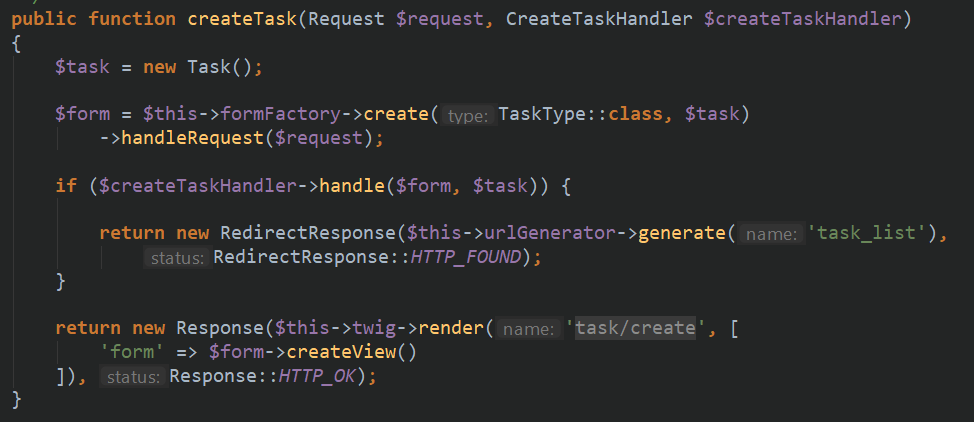
*Réf :* [*https://symfony.com/doc/3.4/controller.html#the-base-controller-classes-services*](https://symfony.com/doc/3.4/controller.html#the-base-controller-classes-services)

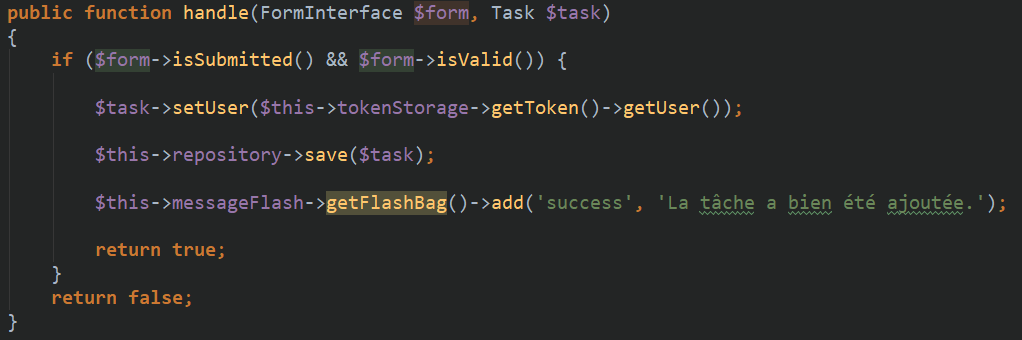
* Seconde optimisation : Les Handler

Il est une bonne pratique de ne pas laissez toute la logique métier dans les contrôleurs. J’ai donc déplacé le traitement des formulaires dans des classes **Handler**



1 - Contrôleur avant modification





- Contrôleur et son handler après modification

**Nouvelles fonctionnalités coté Backend :**

* La commande **CreateAdminCommand** (dossier : AppBundle/Command/) pour facilité l’ajout d’un administrateur à l’application en ligne de commande.

