SensioLabs & moi

Jérémy Jarrié

- Lead Developer chez SensioLabs
- Tech addict, rôliste, amateur de CTF

SensioLabs

- Experts dans l'écosystème Symfony
- Formateurs Symfony & Scrum
- Majoritairement dématérialisée
- Membre du groupe Smile

Sensio Labs

Symfony

Du framework à l'écosystème

SensioLabs

Créateur de Symfony

Qu’est ce que Symfony?

“Symfony is a powerful PHP framework that empowers developers to build scalable, high-performance web applications with reusable components, comprehensive documentation, and a strong community.”

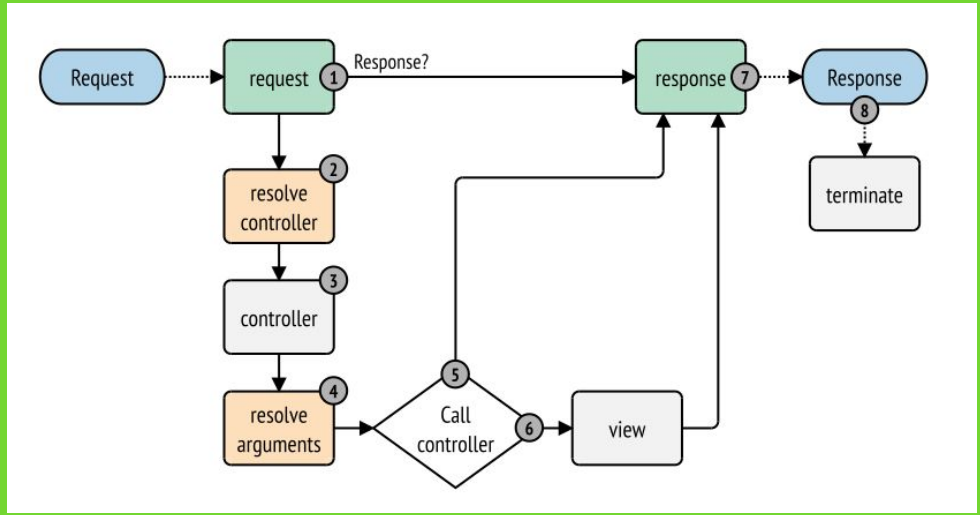
- Open-source, à l’initiative de Fabien Potencier et publié en 2005
- Principalement orienté web, mais pas que
- Collection de 275 composants et packages
- Sert de base à de nombreux autres outils (Librairies, CMS, Frameworks)

Sensio Labs

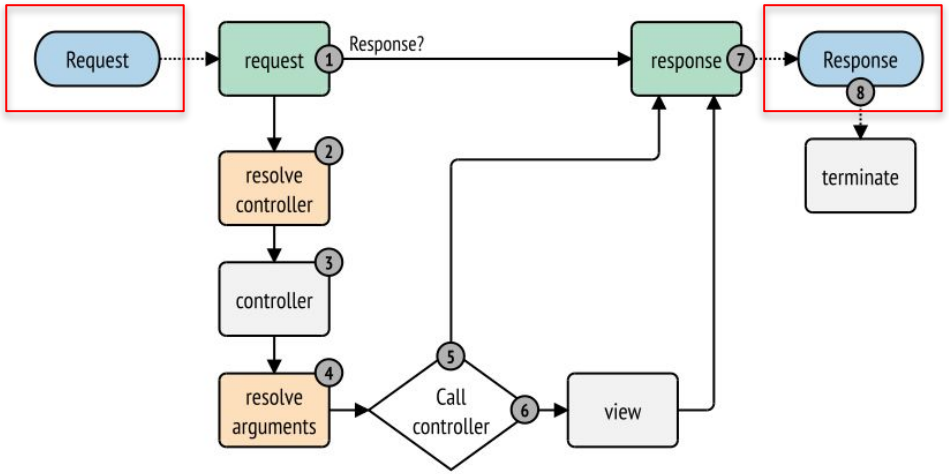
Pourquoi utiliser Symfony?

- Un cadre robuste pour des applications web maintenables
- Respect des standards (PSR)
- Un framework robuste et largement testé
- Une communauté importante et engagée
- Une documentation fournie et largement remplie d'exemples
- Une roadmap, un cycle de release et une promesse de rétro-compatibilité claire en place et appliqués depuis 2015

Le coeur

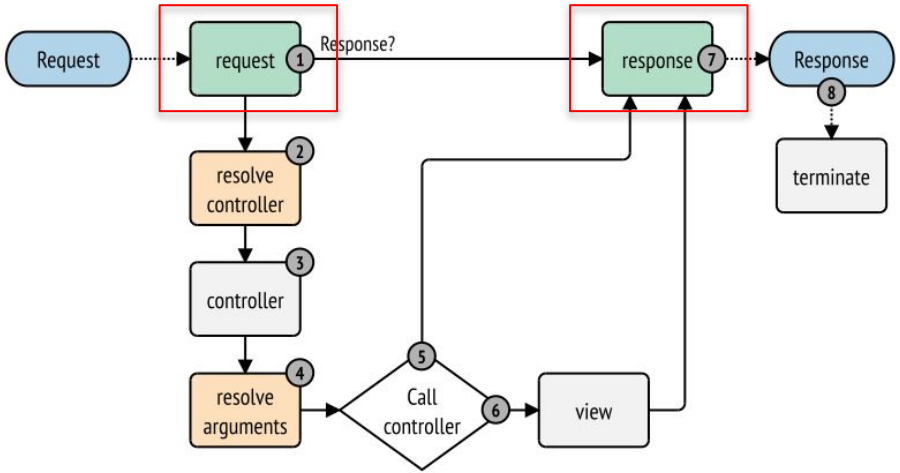

Request
request
1
Response?
response
7
Response
8
terminate
2
resolve controller
3
controller
4
resolve arguments
5
Call controller
6
view

HttpFoundation



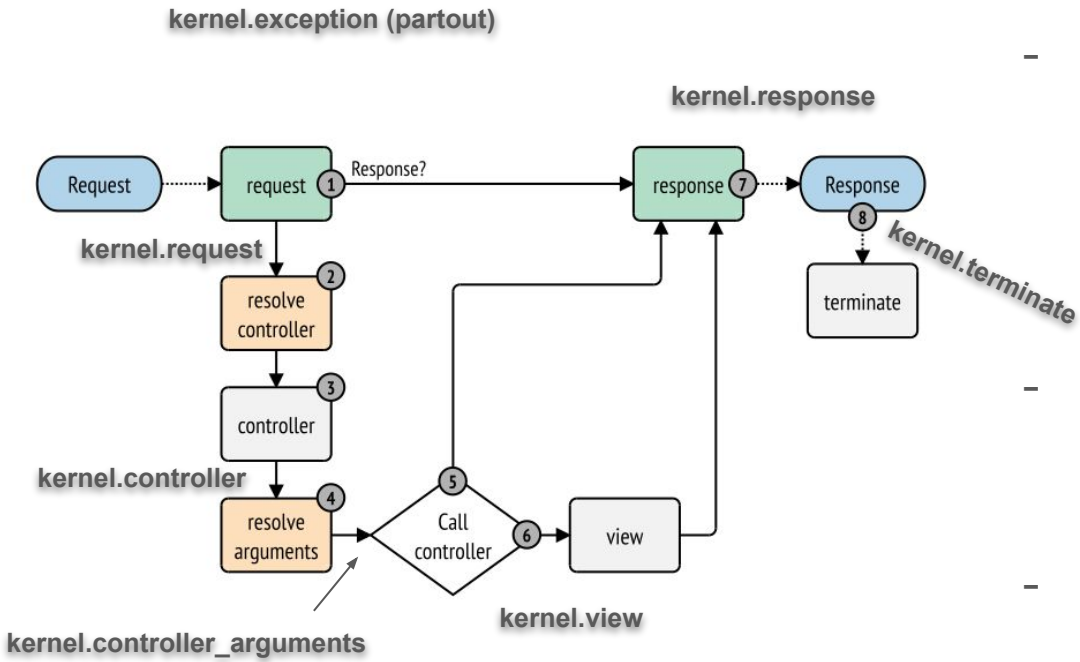
- Défini une couche objet pour la spécification HTTP
- Il convertit la requête HTTP entrante en objet **Request**
- Il va transmettre l'objet **Response** en tant que réponse HTTP
- Il fournit des outils pour manipuler les informations liées à HTTP
- Remplace la manipulation des superglobales PHP

HttpKernel



- Il est responsable du boot de l'application
- Fournit un processus structuré pour transformer une requête en réponse
- Il est le ciment entre la couche de **HttpFoundation** et le reste de l'application
- Il utilise **EventDispatcher** pour faire communiquer les différents composants

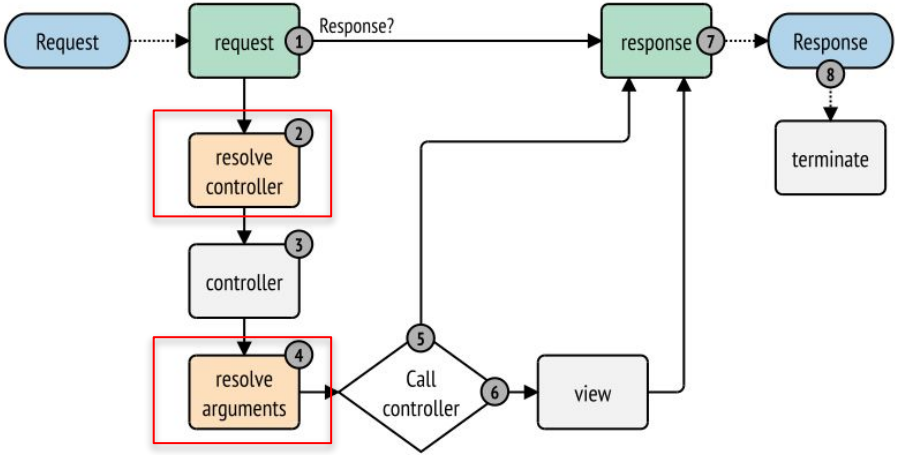
EventDispatcher



- Fournit les outils permettant à l'application de faire communiquer les composants entre eux, en émettant des évènements ou en les écoutant.
- Architecture **Dispatcher/Listener/Subscriber** (**Observer Pattern**)
- Quelques évènements Kernel à connaître

Sensio Labs

Dependency Injection



- Permet de standardiser et centraliser la façon donc les objets sont construits dans l'application
- Résout les dépendances de vos services (**Autowiring**)
- **PSR-11** compatible (**Container**)
- Permet de découpler et faciliter les tests de vos Services

Sensio Labs

Le quotidien

Security

- Fonctionne autour d'une abstraction du concept d'utilisateur, de rôles applicatifs et leur manipulations (**Providers**)
- Features d'authentification (**Firewalls**)
  - Builtins authentification concepts (Form/JSON/HTTP/Access Tokens…)
  - Implémentation custom (**Authenticator**)
- Features d'autorisation (**Access Control**)
  - Couche de contrôle d'accès basé sur le rôle de l'utilisateur (**RBAC**)
  - Couche de contrôle d'accès custom, basé sur des règles métiers (**Voters**/**ABAC**)
- Évènements spécifiques

Form & Validator

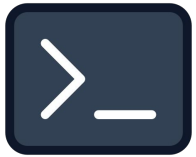
- Couche d'abstraction d'un formulaire
- Définition (**FormBuilder**)
- Affichage (**Twig** form helpers)
- Traitement (**handleRequest**)
- Validation d'un formulaire (**Validator**)
- 40 type de champs intégrés à Symfony
- Possibilité de créer des types de champs custom (**AbstractType**)

Serializer

- Permet de transformer et dé-transformer des représentations d'un objet (en JSON par exemple)
- Plusieurs formats par défaut (JSON, XML, CSV, YAML)
- Hautement configurable
  - Format spécifique (**Normalizer**)
  - Choix de ce qui doit être sérialisé ou non (**Groupe**/**Context**)

Si vous travaillez un jour avec **API Platform**, vous y serez confronté directement ou indirectement!

Console



- Second principal point d'entrée vers l'application après les controlleur http
- Permettre d'écrire des commandes:
  - de type Batch de traitement
  - de maintenance
  - des outils tiers
- Beaucoup de commandes utilitaires déjà présentes de bases:
  - Debug en tout genre (Services, Routes …)
  - Gestion du cache applicatif (Clear, Warmup, …)
  - Commandes spécifiques à certains composants (Make, Messenger …)

HttpClient

- Wrapper autour de plusieurs implémentations de clients HTTP (**cURL**, **PHP streams**)
- Fourni une implémentation compatible **PSR-17** et **PSR-18**
- Permet de lancer des requêtes non bloquantes, tant qu'on n'accède pas à la réponse
- Possibilité de créer des instances associées à un contexte (**Scoped client**)
- Multiplexing
- Décoration fournie pour cacher les requêtes/réponses (**CachingHttpClient**)

AssetMapper

- Remplaçant de **Webpack Encore** pour la gestion des assets fronts (JS images, CSS)
- Peu voire aucune configuration requise
- **Versionning** automatique des assets en prod
- Intégré à **Twig**
- Compatible **Symfony UX**
- Se passe totalement de la stack Node/NPM pour fonctionner
- HTTP/2 et HTTP/3 compatible (**Preloading**, **Parrallelism**)

L’écosystème

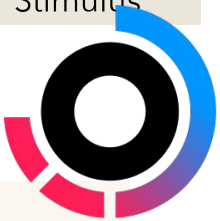
Composer

- Gestionnaire de dépendances pour les projets PHP : installation, suppression et mises à jour
- Gère l'arbre de dépendances pour résoudre correctement les versions à installer
- La configuration des dépendances d'un projet se fait fait dans un fichier **composer.json**
- Une fois résolues, les dépendances sont inscrites dans un fichier **composer.lock** qui peut/doit être partagé
- Permet de vérifier la présence de problème de sécurité liés aux dépendances (**composer audit**)

Symfony Flex

- Plugin composer qui permet de lancer des “**Recipes**”
- Les recipes sont des scripts qui vont inscrire de la configuration ou du code de base associé aux dépendances
- Peut s’adapter aux structures non conventionnelles (**Flex Paths**)
- Permet à la plupart des composants de Symfony d’être plug & play dans les cas d’usages courants

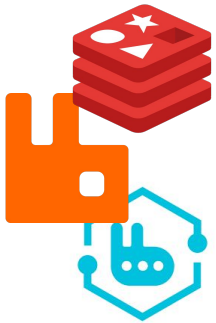
Symfony UX



HOTWIRE

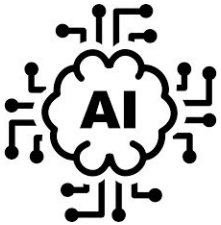
- Intégration de **Stimulus** et **Turbo** dans l'écosystème Symfony
- Fournit une collection de composants front, techniquement tous les composants stimulus sont utilisables
- Intègre les notions de **TwigComponent** & **LiveComponent**
- Intègre **UX Icons**, **UX Map** et d'autres facilitateurs
- Intégration dans **Twig** de composants React, Vuejs ou Svelte

Messenger



- Envoi de **Message**(s) qui peuvent être traités de façon **asynchrone**
- Définition de **Handler** pour traiter les messages
- S'interface avec plusieurs systèmes de transport comme **AMQP** (**RabbitMQ**), **Redis**, **Beanstalkd**, **Doctrine** et d'autres
- Sert de base à certaines implémentations de type **CQRS/Event-driven**
- Système de **Middleware**(s) et d'**Enveloppe**(s) pour enrichir les comportements

Le petit nouveau : Symfony AI

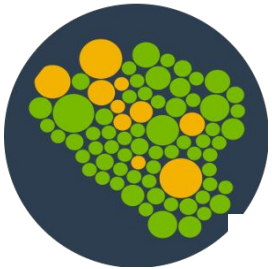


- Ensemble de composants afin d'intégrer de l'IA dans les applications PHP
- Interface unifié pour communiquer avec différentes plateformes d'IA (OpenAI, Anthropic, Gemini …)
- Framework de construction d'agents
- Interface unifié pour communiquer aux agents
- Abstraction de communication avec des stockages vectoriels
- Intégration du SDK MCP officiel

Tester Symfony

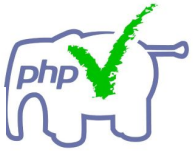
Sensio Labs

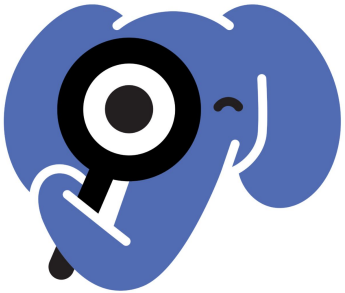
Outils de QA



PHP-CS-Fixer

phparkitect





Phan

Deptract

Sensio Labs

Symfony comme fondation

De nombreux projets et autres framework se basent sur Symfony :

- **API Platform** : framework REST/GraphQL pour fabriquer des APIs
- **Sylius** : Plateforme E-commerce headless
- **Shopware** : Solution E-commerce
- **Drupal 8** : CMS
- **Composer** : Basé sur le composant Console
- **Laravel**
- et beaucoup d’autres!

Merci!