Partie 0 — Le framework Symfony : Déroulement du cours

Bienvenu dans ce cours dédié au framework Symfony.

Symfony va vous aider a structurer tous ce que vous avez abordé précédemment en PHP, notamment la POO et l'architecture pattern MVC.

Le framework a complètement été revu entre sa version 1 et 2, nous sommes à la version 5 actuellement

Voici comment nous allons aborder tout cela ensemble =>

Le framework Symfony

Partie 1 : Présentation du framework

Partie 2 : Les fondements de Symfony

Partie 3 : Les concepts avancées de Symfony

Symfony

Ce module sera divisé en 3 grandes parties :

Prérequis

* Avoir pratiqué le PHP et POO

* Une version PHP 7

Alors, voyons maintenant ensemble les prérequis afin de suivre cette formation :

1/ Connaitre le PHP, la POO et a minima utiliser le terminal.

2/ Avoir au moins la version 7 sur son serveur

3/ Votre IDE favori :: Est ce que c'est bon pour tout le monde ?

Donc c'est parti, commençons ce cours avec le 1er chapitre =>

Partie 1 — cours 1:
Présentation de Symfony

Présentation de Symfony

Symfony aujourd’hui

- Framework PHP open source, français et soutenu par Sensiolabs.
- Framework = cadre de travail structuré, adaptable et « maintenable » facilement.
- Communauté internationale de développeurs et utilisateurs prospère.
- De nombreux « Components » disponibles pour vos applications.

Qu’est-ce qu’un framework ?

« Cadre de travail » => ensemble de composants qui sert de base et de structure pour la création d’un logiciel. Il fournit des outils complets aux développeurs, pour commencer a travailler sur le développement de leurs solutions.

Symfony est un framework PHP stable et reconnu par la communauté des développeurs.

Présentation de Symfony

Le versionnement Sémantique (SemVer)

- SemVer est un système à 3 composants au format x.y.z où :
  - x représente la version dite majeure
  - y représente la version dite mineure
  - z représente la version dite corrective

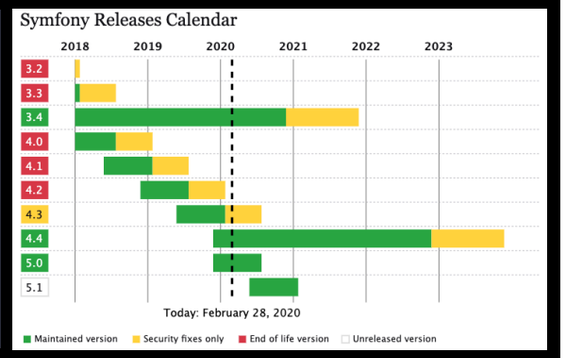
Latest Stable Release	Latest Long-Term Support Release
5.0.4	4.4.4

Le versionnement sémantique est une méthode universelle de présenter l’évolution des logiciels et projets de développement. Il clarifie et ordonne les avancées et autres mises à jour

Présentation de Symfony

Les versions de Symfony

- La stratégie de versioning de Symfony :
  - 1 version mineure tous les 6 mois (Mai et Novembre)
  - 1 version majeure tous les 2 ans
  - La LTS (Latest Long-Term Support)



- Tableau des releases/timeline : https://symfony.com/releases

Nous allons travailler avec la version 4 de Symfony :

=>

Présentation de Symfony

Installation de Symfony 4

- ==> Créer un nouveau projet via Composer (le gestionnaire de dépendance)

```
composer create-project symfony/skeleton my-project
```

- Officiellement, il existe deux squelettes applicatifs :
  - skeleton: minimaliste pour démarrer un projet PHP. Ex: API, applications en ligne de commande…
  - website-skeleton: recommandé pour faire des projets web.

Composer est un outil pour gérer les dépendances en PHP. Les dépendances, dans un projet, ce sont toutes les bibliothèques dont votre projet dépend pour fonctionner. Composer installera tout ce dont vous avez besoin, pour le bon fonctionnement de votre projet Symfony.

Depuis la version 4 du framework Symfony, il n'y a plus vraiment de distribution officielle.

La vision de l'équipe qui maintient le projet, c'est de fournir des squelettes applicatifs légers, et de laisser la responsabilité aux développeurs de décider quelles dépendances sont nécessaires dans leurs applications.

composer create-project symfony/website-skeleton:^4.4 my_project_name =>

Présentation de Symfony

Démarrer votre projet

- ==> lancer le projet

```
cd my-project
composer require server --dev
php bin/console server:run
```

- ==> Ouvrir le projet en local dans le navigateur : http://localhost:8000/

« composer require server —dev », a faire 1 seul fois s'il n'est pas installé dans votre projet.

Si le port n'est pas occupé, l'application sera alors disponible à cette adresse : http://localhost:8000/ . Pour stopper ce serveur local, utilisez la commande Ctrl + C dans votre invite de commande.

Par défaut, puisque vous n'avez aucun contrôleur ni aucune route configurée, le framework vous présente une page avec un lien vers la documentation officielle.

=>

Présentation de Symfony

Symfony Flex

- Flex est un nouveau moyen pour gérer des applications Symfony
- Il automatise les tâches de configuration etc, des composants se trouvant sur symfony.sh.
- Ces descriptions de configuration sont appelées des « recettes » (recipes).
- Flex facilite donc l'installation, la configuration et la désinstallation de bibliothèques tiers.
- Disponible depuis Symfony 3.3+ (Par défaut depuis Symfony 4.0)

Symfony Flex est la nouvelle façon d'installer et de gérer les applications Symfony. En effet, ce n'est pas une nouvelle version de Symfony, mais un outil (ou une surcouche) qui remplace et améliore le programme d'installation de Symfony. Symfony Flex peut marcher sur un projet Symfony 3.3 mais devient obligatoire sur un projet Symfony 4.

En d'autres termes, Symfony Flex est un plugin Composer qui modifie le comportement des commandes require, update et remove.

Lors de l'installation ou de la suppression de dépendances dans une application compatible Flex, Symfony peut effectuer des tâches avant et après l'exécution des tâches Composer.

Lorsque Symfony Flex est installé dans l'application et que vous exécutez composer require, l'application envoie une requête au serveur Symfony Flex avant d'essayer d'installer le paquet avec Composer :

S'il n'y a aucune information sur ce paquet, le serveur Flex ne renvoie rien et l'installation du paquet suit la procédure habituelle basée sur Composer;

S’il existe des informations spéciales sur ce package, Flex le renvoie dans un fichier appelé “recette” et l’application l’utilise pour décider du package à installer et des tâches automatisées à exécuter après l’installation.

Une recette Symfony est décrite dans un fichier de configuration « manifest.json ».

Quand vous installez un nouveau composant Symfony, une recette va s’appliquer et configurer automatiquement votre application Symfony.

Exemple : Lorsque vous installez Doctrine la recette va créer pour vous automatiquement une variable d’environnement « database_url » pour la connexion à votre base de données.

https://symfony.sh/ et https://github.com/symfony/recipes/blob/master/doctrine/doctrine-bundle/1.6/manifest.json

Présentation de Symfony

L’architecture des fichiers

```
.
├── bin/
├── composer.json
├── composer.lock
├── config/
├── phpunit.xml.dist
├── public/
├── src/
├── symfony.lock
├── templates/
├── tests/
├── translations/
├── var/
└── vendor/

9 directories, 4 files
```

- bin/ : les executables du projet (console symfony etc.)
- config/ : configuration des routes, services, packages etc.
- public/ : répertoire racine du projet. Accessible publiquement.
- src/ : le code PHP (Vos classes etc. sont ici).
- templates/ : tous les template Twig (vues) ici.
- tests/ : les tests unitaires, d'intégration et d’interfaces.
- var/ : fichiers de cache et log.
- vendor/ : autoloader et autres dépendances installées via Composer.

Présentation de Symfony

Le contrôleur frontal

- Point d’entrée de votre application
- index.php se situant dans le dossier public/.
- C'est le fichier par lequel passent toutes vos pages.
- Il appel simplement le noyau (Kernel) de Symfony, pour déléguer la gestion de la requête au Kernel.

On l’a utilisé dans notre projet MVC avec les query-string : index.php?page=blog

Présentation de Symfony

Les environnements

- Il existe 2 environnements de travail :
  - « prod » est destiné aux visiteurs (rapide et n'affiche pas les messages erreurs)
  - « dev » est destiné aux développeurs (affichent les informations pour le débug)
- L'environnement est paramétrable dans le fichier .env

La page de bienvenue que nous avons vu plus tôt n'est en fait disponible que dans l'environnement de développement « dev ».

C'est bien évidemment une page que l'on ne veut pas afficher à nos futurs visiteurs ! On a donc ici une erreur 500, toute blanche parce que mon serveur web (Apache) n'est pas configuré pour afficher autre chose => mais on pourrait créer une page pour les pages « non trouvées ».

Présentation de Symfony

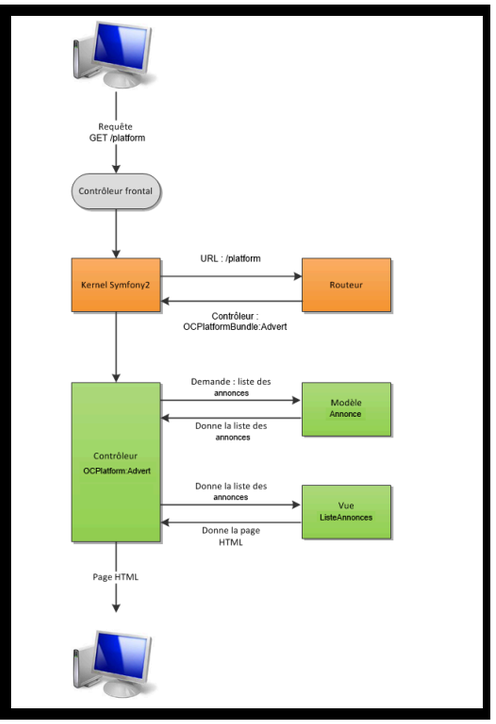
Rappel sur l’architecture MVC

- Symfony utilise l’architecture MVC
- Controleur : vérifier les données et générer la réponse HTTP (vers la vue).
- Modèle : gérer les données et contenus dont a besoin le contrôleur (via requêtes SQL par exemple).
- Vue : afficher les pages (les balises HTML etc.)

On vient de voir comment sont organisés les fichiers de Symfony. Maintenant, il s'agit de comprendre comment s'organise l'exécution du code au sein de Symfony.

Présentation de Symfony

Parcours d’une requête dans Symfony



1/ Le visiteur demande la page
2/ Le Contrôleur frontal reçoit la requête, charge le Kernel et la lui transmet
3/ Le Kernel demande au Routeur quel Contrôleur exécuter pour la demande du visiteur
4/ Le Kernel exécuté ce Contrôleur
5/ Le Contrôleur demande au Modèle les données dont il a besoin
6/ Le Contrôleur vérifie et traite les données avant de les envoyer à la Vue

Présentation de Symfony

Résumé

- Symfony peut s'installer simplement avec composer.
- Symfony est organisé en 7 répertoires principaux (bin, config, public, src, templates, var et vendor)
- Nous travaillerons principalement dans src, templates et config.
- On peut configurer 2 environnements de travail (dev et prod)
- Symfony utilise l'architecture MVC

Résumé

Vidéo interview du créateur de Symfony :

https://openclassrooms.com/fr/courses/3619856-developpez-votre-site-web-avec-le-framework-symfony/3620442-aux-origines-de-symfony-interview#/id/video_Player_1

Est-ce que vous avez des questions ?