Démarrer un projet avec Symfony

Le Framework Symfony est un outil avancé permettant aux développeurs PHP de gagner un temps considérable lors du développement d'un nouveau site web.

Plan du chapitre:

- Introduction
- Installation du framework Symfony
- Déploiement de notre application
- Configuration Apache
- Erreurs fréquentes

Introduction

http://www.symfony-project.org/ est un framework PHP pour vos projets web développés en PHP5.

Mais qu'est-ce qu'un framework ?

Pour faire simple, un framework est un ensemble d'outils permettant de ne pas réinventer la roue. Il propose plusieurs modules solides, optimisés, testés par plusieurs développeurs et donc fiables.

Un framework permet aussi d'imposer des normes et standards de développement qui permettent une réutilisation du code dans vos autres applications.
Toujours grâce à ces standards, il sera bien plus simple pour un autre développeur de prendre la suite du développement d'un site développé avec Symfony.

Tous les composants d'un framework sont faits pour fonctionner les uns avec les autres. Assemblés, ils forment une architecture efficace permettant de créer rapidement une application (Web PHP dans le cas de Symfony).

Il existe plusieurs Frameworks PHP libres comme Symfony: Zend, CakePHP...
Toutefois, c'est Symfony que j'ai retenu, du fait de son architecture MVC très poussée, sa facilité de développement, sa philosophie, sa communauté ...

Le Framework Symfony

Symfony a été lancé en octobre 2005 par Fabien Potencier, pour la société française http://www.sensiolabs.com/fr.

Il utilise plusieurs concepts et design pattern permettant de ne pas réinventer la roue:

- Structure MVC: Modèle Vue Contrôleur.
- ORM: Mapping Objet Relationnel (Doctrine ou Propel)
- Tests unitaires et fonctionnels
- Gestion des environnements: Développement, production, tests ...
- Sécurité: XSS ...
- Outils de debugging
- Administration auto générée
- Internationalisation
- Extensions: Gestion des plugins

Vous êtes peut être un peu perdu dans ce vocabulaire technique, mais je tenterai de vous faire comprendre l'essentiel tout au long du tutorial.

Pour les plus réfractaires, sachez que des sites à très forte charge utilisent le framework, c'est le cas de dailymotion ou Yahoo!

Installation du framework Symfony

Nous allons pouvoir commencer à installer le framework sur notre environnement de développement web.

Il existe plusieurs façons de l'installer et sur différents OS.
Dans le tutorial, nous allons voir comment l'installer sur un serveur web linux déjà installé (Debian/Ubuntu).

Si vous souhaitez savoir comment installer un serveur web sous linux, vous pouvez suivre le tutorial de la Ferme du Web: Mettre en place un serveur Web sous debian.

Il est possible d'utiliser Symfony localement avec Wamp ou Mamp en téléchargeant la version Sandbox de Symfony, mais l'objectif du tutorial est de créer un véritable environnement de production.

Si vous n'avez pas de machine permettant de faire serveur, je vous recommande la virtualisation (Machine virtuelle Symfony).

Méthodes d'installation de l'environnement Symfony:

Avec PEAR

Avec SVN

Pré installé avec la machine virtuelle Symfony

Personnellement, j'ai utilisé pear pour me faciliter la vie, mais c'est bien d'utiliser SVN pour garder la version à jour ou faciliter les migrations.

Installation de Symfony avec PEAR

Dans un premier temps, installez pear sur votre distribution si ce n'est pas encore fait:
apt-get install php-pear

Une fois que le pear est installé, il faut découvrir le channel dédié à Symfony: pear channel-discover pear.symfony-project.com

Puis télécharger la dernière version du framework. (Changez le 1.2.7 par la dernière version de Symfony) pear install symfony/symfony-1.2.7

Et voilà normalement Symfony est installé et prêt à utilisation !
Pour vérifier que tout s'est bien passé, tapez la commande suivante: symfony -V

Vous devriez obtenir un message vous indiquant la version du framework installé.

Si vous rencontrez des problèmes d'installation, expliquez votre problème avec l'erreur dans le forum (topic dédié).

Si vous souhaitez installer Symfony d'une autre manière, allez faire un tour sur la documentation officielle.

Déploiement de notre application

Afin d'avoir les mêmes répertoires que pour le tutorial, je vous conseille de créer un répertoire watchmydesk/ dans le homedir de votre utilisateur linux.

Maintenant que tout est installé, nous allons générer l'application.
Toujours en ligne de commande, un projet web sous Symfony se créé de la manière suivante: symfony generate:project watchmydesk

Où watchmydesk est le nom du projet que l'on va créer.
Les fichiers du projet sont désormais créés.

Description des fichiers

Regardons brièvement l'architecture générée:

Code:
apps/ Contient les différentes applications du projet
cache/ Contient tous les fichiers de cache
config/ Fichiers de configuration du projet
lib/ Répertoire où sont stockés les librairies, classes PHP, les modèles...
log/ Traces et debug du framework
plugins/ Plugins installés dans le projet
test/ Répertoire contenant les tests unitaires / fonctionnels
web/ Le répertoire web sur lequel pointe notre domaine.

Création de deux applications

Symfony permet de créer plusieurs applications dans un même projet.
Dans notre cas, nous aurons deux applications:

Application frontend: La partie publique du site

Application backend: La partie back office

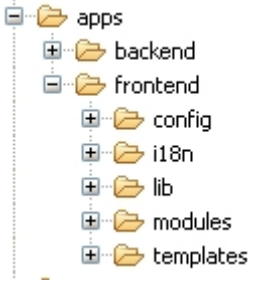
Vous l'aurez compris, il faut encore utiliser une ligne de commande pour générer les deux applications.

Créons l'application frontend: symfony generate:app --escaping-strategy=on --csrf-secret=UniqueSecret frontend

Remplacez UniqueSecret par une chaine de caractères secrète qui permettra de contrer nativement les attaques CSRF.

Même chose pour l'application backend: symfony generate:app --escaping-strategy=on --csrf-secret=UniqueSecret backend

Plusieurs nouveaux fichiers ont été créés dans le répertoire apps/:



Code:
config/ Les fichiers de configuration
i18n/ Les fichiers de langue
lib/ Les librairies et classes
modules/ Les modules contenant le code de notre application
templates/ Le layout principal

Configuration Apache

Maintenant que tous les fichiers de base sont créés, nous allons configurer notre serveur Apache afin d'accéder directement à notre application web.

Nom de domaine

Afin d'accéder facilement à notre projet, nous allons créer un domaine virtuel: dev.watchmydesk.com

Comme vous n'êtes pas le possesseur de ce domaine, nous allons feinter et ajouter l'entrée du domaine dans nos DNS.

Configuration de votre OS client

Sous Windows:
Modifiez le fichier C:/WINDOWS/system32/drivers/etc/hosts
Puis ajoutez la ligne suivante: IP_DE_VOTRE_SERVEUR dev.watchmydesk.com

Sous linux / Mac:
Modifiez le fichier /etc/hosts
Puis ajoutez la ligne suivante: IP_DE_VOTRE_SERVEUR dev.watchmydesk.com

Sauvegardez le fichier et quittez tous vos navigateurs web ouverts.

Configuration du serveur

Modifiez le fichier /etc/hosts
Puis ajoutez la ligne suivante: 127.0.0.1 dev.watchmydesk.com

Création du vHost apache

Créez un nouveau fichier virtual host pour apache2: nano /etc/apache2/sites-enabled/watchmydesk

Puis copiez y ceci: Code:

```
<VirtualHost *>
  ServerName dev.watchmydesk.com
  DocumentRoot "/home/djo/public/watchmydesk/web/"
  DirectoryIndex index.php
  <Directory "/home/djo/public/watchmydesk/web/">
    AllowOverride All
    Allow from All
  </Directory>

   Alias /sf /usr/share/php/data/symfony/web/sf/
  <Directory "/usr/share/php/data/symfony/web/sf/">
    AllowOverride All
    Allow from All
  </Directory>
</VirtualHost>
```

Il faut bien sûr que vous adaptiez les chemins à votre environnement.

Enregistrez, quittez (CTRL + X) et on reboot apache et on test !

/etc/init.d/apache2 restart

Ca marche !

Erreurs fréquentes

Si vous rencontrez des problèmes dans cette première partie du tutorial, jetez un coup d'oeil à ces erreurs fréquemment rencontrées:

Les images et le style de la page ne s'affichent pas:
Dans le cas où vous n'auriez pas les images, vérifiez de chemin de l'Alias /sf du vhost:
/usr/share/php/data/symfony/web/sf/
Regardez si le chemin pointe bien vers le répertoire web/sf/ du framework Symfony, sinon corrigez manuellement.
Dans certaines versions de debian, le chemin correct est celui-ci:
/usr/share/php/symfony/data/web/sf

Le nom de domaine n'est pas trouvé:
Vérifiez d'avoir bien entré les lignes DNS dans vos fichiers hosts client et serveur.
Quittez bien tous les navigateurs web (même fenêtre de download) puis relancé le.

La base de données et le projet: Watch My Desk

Dans cette deuxième partie, nous allons définir le projet Watch My Desk plus en détail, sa base de données ...
Ensuite, nous verrons comment configurer correctement Symfony avec notre base de données et générer le schéma de la BDD à partir de nos tables MySQL précédemment créées.

Plan du chapitre:

- Le projet Watch My Desk
- La base de données MySQL
- Générer le schéma de la BDD sous Symfony

Le projet Watch My Desk

Afin d'illustrer le tutorial, j'ai pensé à créer un site qui soit à la fois fun, pas complètement inutile et suffisamment complexe pour voir un peu tous les points spécifiques du framework Symfony.

Mise en contexte

Tout bon développeur, graphiste ou acteur du web a souvent du matériel informatique à la pointe de la technologie.

(http://www.flickr.com/photos/crouchingdonkey/sets/72157594277366817/)

En général votre espace de travail, à savoir votre bureau, est un lieu qui vous tient très à coeur et que vous tentez (certains au moins !) de garder ergonomique, propre et design comme lorsque vous développez ou designez.

Watch My Desk

Watch My Desk est un portail communautaire présentant les bureaux des acteurs du web.

Chaque internaute aura la possibilité de mettre en ligne la description de son bureau, accompagné par une ou plusieurs photos mettant en évidence son matériel et son espace de travail.
Chaque internaute aura la possibilité de consulter ces "fiches bureaux" et d'y poster un commentaire, de le noter etc.
Sur la homepage, un classement des bureaux les plus populaires et plus récents sera affiché.

(http://www.flickr.com/photos/karizon/sets/72157619259207115/)

Ce projet est parti d'un passe temps personnel. En effet j'aime voir sur quoi travail telle ou telle personne derrière son écran. Et je pense ne pas être le seul !

Les modules et fonctionnalités

- Frontend:

Index:

- 5 Derniers bureaux postés (images miniatures) description...
- 5 Bureaux les plus populaires (images miniatures) description...
- Accès rapides pour: S'inscrire, s'identifier, retrouver son mdp.

Liste des bureaux:
Simple liste paginable avec les miniatures des bureaux et une courte description

Classements: Par titre, par auteur, par popularité, par date d'ajout

Détail des bureaux:
Lors du clic sur un bureau, on affiche le détail de ce dernier:
Titre

Photo principale en grand

Liste des autres photos en miniatures: Au clic, la photo se charge à la place de la principale

Description de l'auteur

Auteur

Date d'ajout

Note: assez visible (et permettre de voter juste à coté)

Classement: Position suivant la note et nbr votants

Commentaires: En bas

Permettre d'ajouter un commentaire

S'inscrire:
Formulaire d'inscription ultra simplifié: Login / Pass / Email / Pays

Poster un bureau (être identifié)
Formulaire complet:

Titre

Multi Upload pour les photos

Choix de l'image principale

Description

Voir le profil d'un membre
Ses commentaires

Ses bureaux

son Pays

- Backend

Gestion des membres
Liste

Ajout

Modification

Suppression

Infos sur un membre: Dates d'inscription, IP d'inscription, niveau, statistiques: Coms + bureaux + notes

Gestion des commentaires
Liste

Suppression

Gestion des bureaux
Liste des bureaux en attente de validation

Valider la mise en ligne d'un bureau

Liste des bureaux validés

Ajout

Modification

Suppression

Statistiques
Nombre de bureaux: A valider / Publiés

Nombre de commentaires

Nombre de membres

Voilà en gros les fonctionnalités qui devront être présentes sur notre site. Il se peut que l'on en supprime en court de route, si le temps nous manque, mais voici la base que je me suis fixé.

La base de données MySQL

Maintenant que l'on a vu quelles fonctionnalités seront présentes dans notre projet, passons à la base de données.

Avec Symfony, vous avez deux possibilités pour gérer votre base de données:

Soit vous créer le SQL dans votre base puis vous générez le schéma de la BDD

Soit vous créer manuellement le schéma de la base avec un fichier de configuration YML puis vous générez le SQL

En ce qui me concerne, je préfère créer la base de données visuellement avec MySQL Workbench pour bien réfléchir à tous les champs et liaisons, générer le SQL qui en découle, puis générer le schéma de la BDD Symfony.

MySQL Workbench

Si vous ne connaissez pas encore ce merveilleux outil gratuit, je vous invite à consulter ce billet.

MySQL Workbench à l'avantage de créer toutes les contraintes MySQL comme les clés primaires, étrangères, index etc. Très utiles par la suite dans notre projet Symfony.

Le schéma de notre base de données ne va pas être très compliqué.

Membres
Un membre peut poster un ou plusieurs bureaux
Un membre peut poster un ou plusieurs commentaires
Un membre peut voter pour un ou plusieurs bureaux

Votes
Un vote est associé à un membre et à un bureau donné
Un vote comporte une note sur 20

Commentaires
Un commentaire est associé à un membre et un bureau donné
Il comporte un état: 1 = Publié / 0 = en attente de publication / 2 = Modéré

Bureaux
Un bureau est posté par un membre
Un bureau peut avoir plusieurs commentaires
Un bureau est soumis aux votes des membres, ce qui lui donne une note finale stockée dans la table
Un bureau est classé, en fonction de ses votes et sa note
Un bureau dispose d'une photo principale et peut avoir d'autres photos complémentaires

Photos
Une photo est associée à un bureau donné
Chaque photo est stockée sur le serveur, on conserve son URL dans la table

Voilà en ce qui concerne le schéma de la BDD du projet.
Vous pouvez maintenant générer le SQL du schéma:

Sauvegardez votre fichier SQL exporté.

Créez ensuite un nouveau compte MySQL avec une base de données vierge sur laquelle l'utilisateur MySQL aura tous les droits.

Exemple avec phpMyAdmin: Privilèges > Ajouter un utilisateur

Il ne vous reste plus qu'à importer le SQL des tables que l'on a créé avec MySQL Workbench.
Vous pouvez télécharger le SQL: Sur la Ferme du Web

Vérifiez bien que toutes vos tables aient bien été créées.

Nous sommes prêts pour passer à la configuration de notre projet Symfony !

Générer le schéma de la BDD sous Symfony

Repassons à notre projet Symfony, vous pouvez réouvrir votre console SSH sur votre serveur.

Un ORM pour gérer la base de données: Propel ou Doctrine

Symfony utilise un ORM pour gérer les requêtes et accès en base de données.
ORM pour Object-Relational Mapping.
Pour faire simple, un ORM permet de créer des objets à partir des tables d'une base de données fonctionnelle. C'est en quelque sorte une couche entre les objets d'un langage et les tables d'une base de données.

Symfony supporte deux ORM PHP5: Propel et Doctrine.
Toutefois, l'avenir de Propel est limité, contrairement à celui de Doctrine.
Jetez un coup d'oeil à ce billet comparatif entre les deux ORMs Doctrine et Propel.

Dans ce tutorial, nous prendront en charge l'ORM Doctrine.

Configurer les accès à la base de données

Commençons par configurer les accès à notre base de données dans symfony.
Pour le faire, vous pouvez utiliser la commande suivante:
symfony configure:database --name=doctrine --class=sfDoctrineDatabase "mysql:host=localhost;dbname=watchmydesk" watchmydesk PASS_USER_MYSQL_WATCHMYDESK

(Commande sur une ligne)

En remplaçant bien les valeurs par les votre. (Nom de la base, utilisateur MySQL et mot de passe de la base).

Jetons un oeil au fichier de configuration de la base de données:
nano config/databases.yml

Vous devriez avoir quelque chose comme ceci: Code:

```
dev:
  propel:
    param:
      classname: DebugPDO
test:
  propel:
    param:
      classname: DebugPDO
all:
  propel:
    class: sfPropelDatabase
    param:
      classname: PropelPDO
      dsn: "mysql:dbname=watchmydesk;host=localhost"
      username: root
      password: null
      encoding: utf8
      persistent: true
      pooling: true
  doctrine:
    class: sfDoctrineDatabase
    param:
      dsn: "mysql:host=localhost;dbname=watchmydesk"
      username: watchmydesk
      password: PASS_USER_MYSQL_WATCHMYDESK
```

Tous les fichiers de configuration dans Symfony, utilisent un format favorisant la lecture, le YAML, fichiers ayant pour extension: .yml
Vous l'aurez vite compris, le YAML se structure hiérarchiquement en utilisant un double espace (Attention à ne pas mettre de tabulation ! Seulement un double espace.)

Pour revenir à notre fichier de configuration database.yml, vous pouvez voir qu'il reste des instructions dédiées à Propel, qui est l'ORM par défaut dans Symfony 1.2
Nous allons donc devoir faire du ménage, comme ceci: Code:

```
dev:

test:
```

```
all:
  doctrine:
    class: sfDoctrineDatabase
    param:
      dsn: "mysql:host=localhost;dbname=watchmydesk"
      username: watchmydesk
      password: PASS_USER_MYSQL_WATCHMYDESK
```

Sauvegardez le fichier une fois les modifications faites.
Pour le moment, nous ne pouvons toujours pas utiliser Doctrine car Propel est toujours activé, il faut donc procéder à quelques modifications supplémentaires pour l'éradiquer complètement.
nano config/ProjectConfiguration.class.php

Modifiez ensuite la ligne suivante: Code:
$this->enableAllPluginsExcept(array("sfDoctrinePlugin", "sfCompat10Plugin"));

Comme ceci: Code:
$this->enableAllPluginsExcept(array("sfPropelPlugin", "sfCompat10Plugin"));

Ainsi, nous désactivons le module Propel et activons doctrine.

Quelques fichiers spécifiques à Propel demeurent encore dans notre projet, supprimons-les:
rm -rf config/propel.ini config/schema.yml web/sfPropelPlugin

Et créons le répertoire dédié à doctrine qui stockera le schema.yml généré:
mkdir config/doctrine

Pour que Symfony prenne en compte nos modifications, nous allons supprimer le cache:
symfony cc

Et installer le nouveau plugin doctrine à l'aide de cette commande:
symfony plugin:publish-assets

Génération du schéma de la BDD dans Symfony

Symfony doit connaitre à tout moment les relations entre vos tables, les champs, contraintes etc. afin de pouvoir utiliser correctement les fonctions de Doctrine.

Symfony stocke un fichier schema.yml, encore un fichier de configuration YAML, qui représente votre base de données.
Comme nous avons déjà créé notre SQL, nous allons générer ce fichier à partir de notre base.
Mais il aurait été possible de créer le fichier schema.yml à la main puis générer la base de données MySQL.

On génère le schéma: symfony doctrine:build-schema

Je vous conseille d'aller faire un tour dans le fichier pour comprendre comment cela fonctionne:
nano config/doctrine/schema.yml

Voilà pour cette deuxième partie, nous attaquerons le MVC dès lundi prochain !