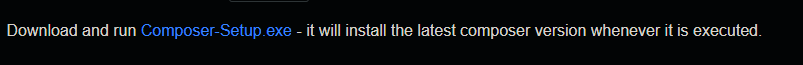
SYMFONY

COMPOSER : Dependency Manager for PHP.

**Installation composer :**

<https://getcomposer.org/download/>

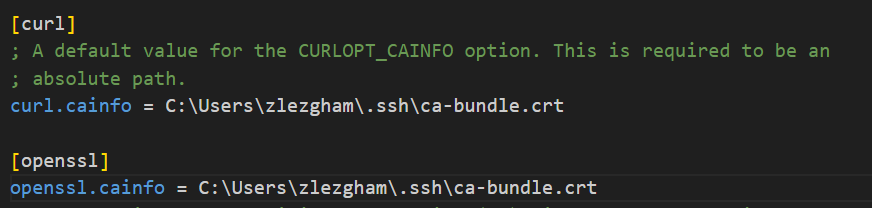


Go to :

<https://raw.githubusercontent.com/bagder/ca-bundle/master/ca-bundle.crt>

download the certificate

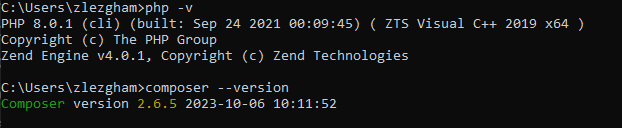
go to your php file, and into php.ini, add the following line :



curl.cainfo = C:\Users\zlezgham\.ssh\ca-bundle.crt

openssl.cainfo = C:\Users\zlezgham\.ssh\ca-bundle.crt

Path to your certificate.



Php -v => to check your php version.

Composer –version => to check your php version.

**Install symfony :**

Officiellement, il existe deux squelettes applicatifs :

* Skeleton : le squelette minimaliste pour démarrer un projet PHP, il est recommandé pour des applications en ligne de commande, par exemple.
* Website-skeleton : le squelette recommandé pour faire des projets web.

Un projet de démo : Dans ce projet, vous trouverez quelques contrôleurs, quelques vues intégrées avec Bootstrap, quelques formulaires et un espace d'administration minimaliste déjà mis en place.

<https://github.com/symfony/demo>

**Symfony CLI,**

Offre 3 fonctionnalités :

* La création d'applications Symfony à partir des squelettes applicatifs ;
* Un serveur local pour exécuter votre projet sur votre poste ;
* La création d'applications Symfony à partir des squelettes applicatifs ;

L’installation du squelette website skeleton se fait à l’aide de Symfony CLI :

* Symfony new –webapp mon-super-projet

Install symfony CLI

<https://symfony.com/download>

Add the file into your environment variables.

If you have proxy issues, either connect to wifi-guest (no proxy) or configure your proxy.

**Démarrage de l’application**

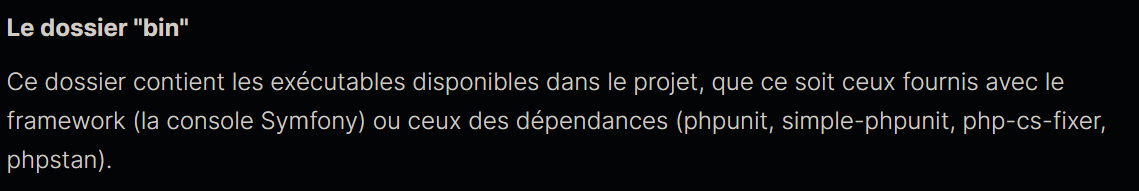
En prod, on utilise un serveur web comme apache ou Nginx, en développement on utilise un serveur local de PHP.

* Cd myprojet
* Symfony server : start

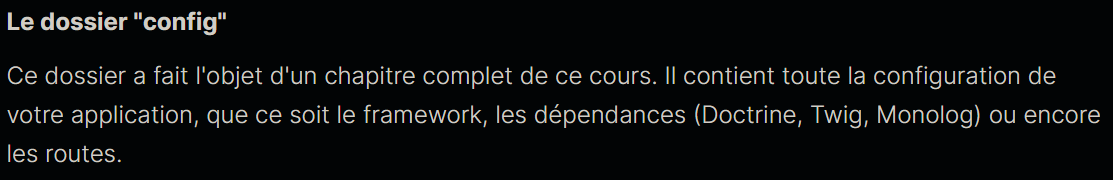
**Structure du Framework Symfony**



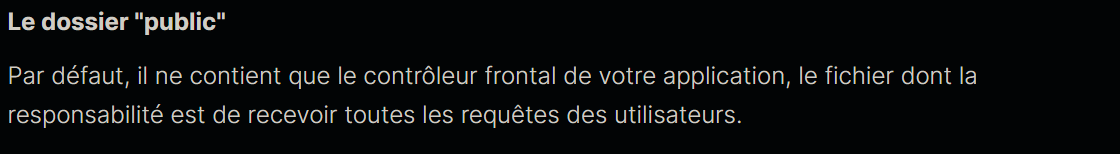
Le dossier bin



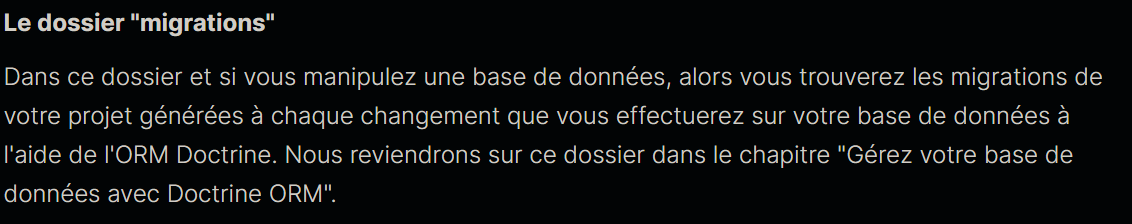
Le dossier config



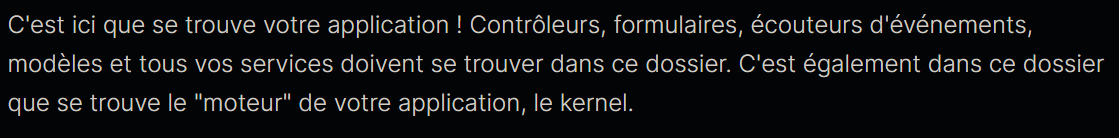
Le dossier public



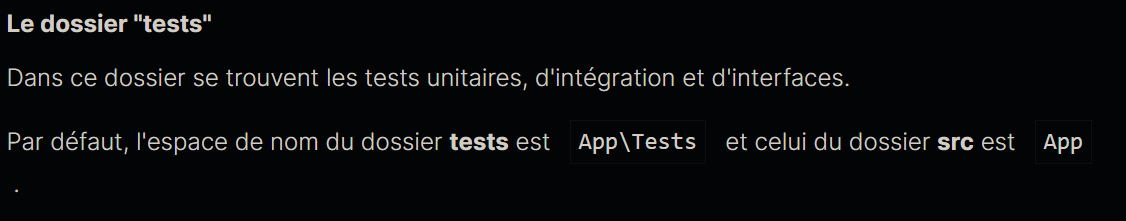
Le dossier migrations



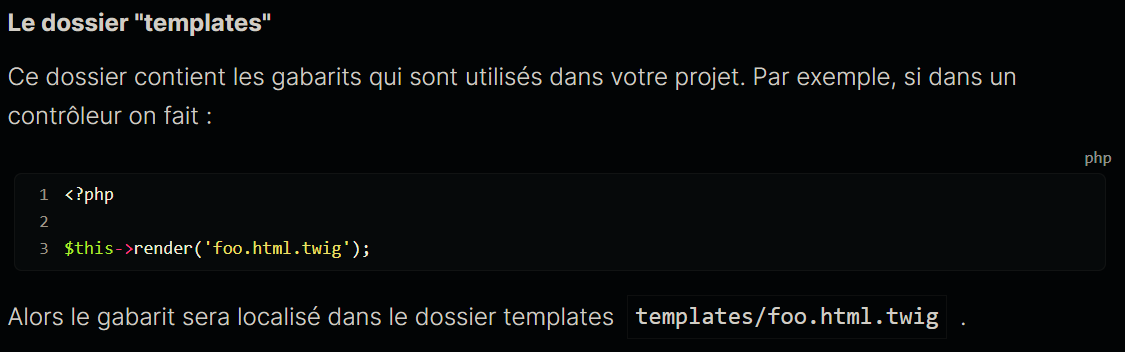
Le dossier src



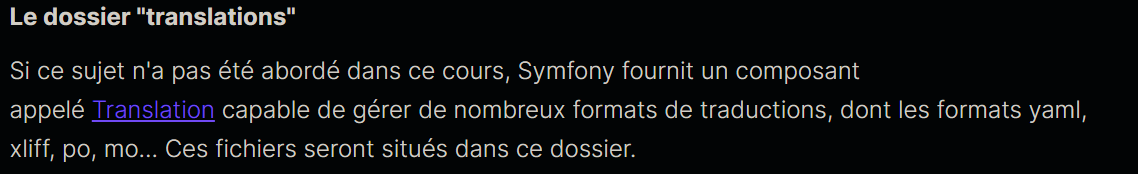
Le dossier tests



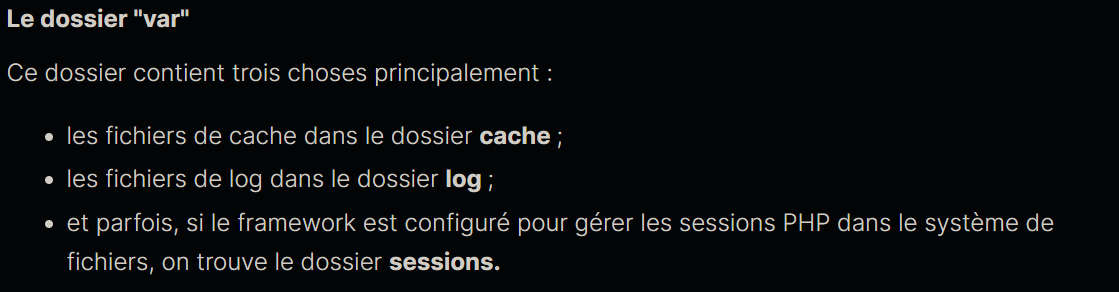
Le dossier templates



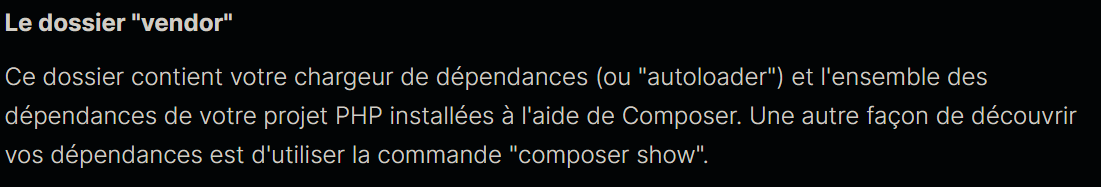
Le dossier translations



Le dossier var

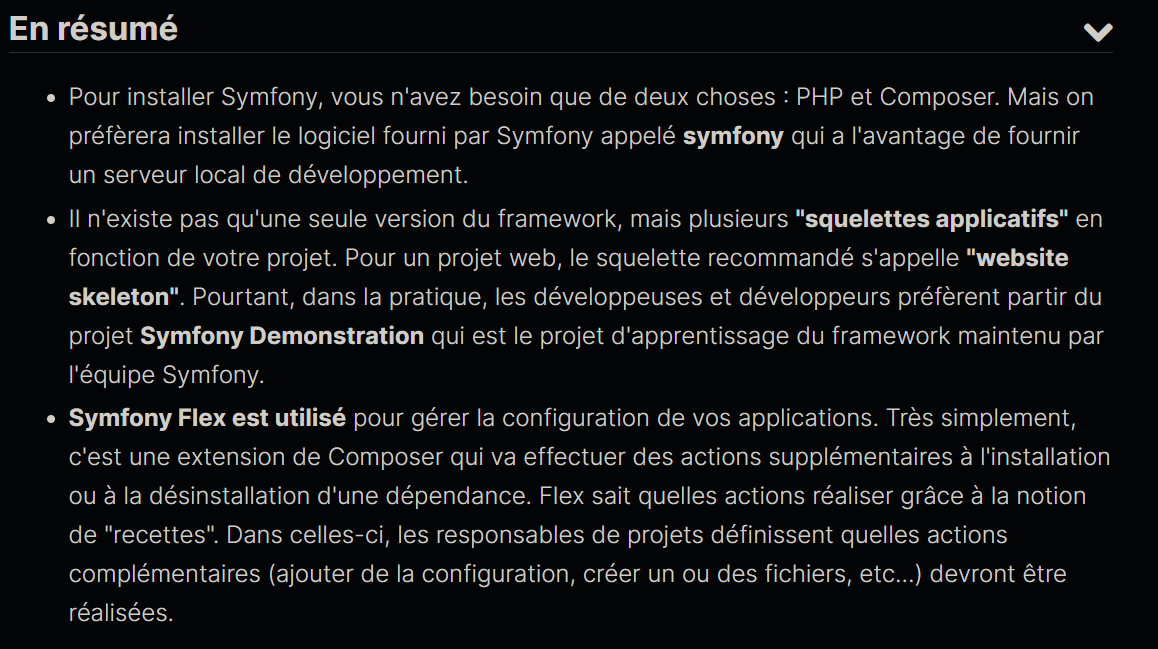
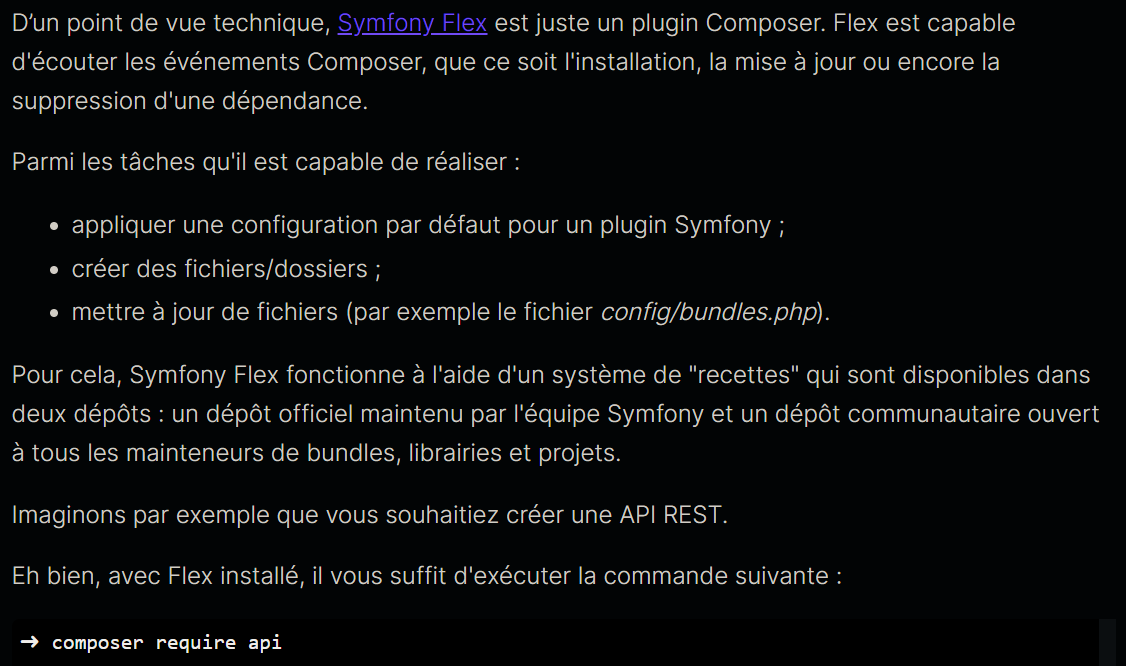


Le dossier vendor



* Composer show ( too see your dependencies)

**Introduction à Symfony Flex**

****

**Requêtes et réponses en Symfony**

**La classe request :**

La classe [Request](https://symfony.com/doc/current/introduction/http_fundamentals.html" \l "symfony-request-object" \t "_blank) permet de centraliser l'accès à toutes les super variables de PHP en une seule classe utilitaire



**La classe Response**

La classe [Response](https://symfony.com/doc/current/introduction/http_fundamentals.html" \l "symfony-response-object" \t "_blank) permet de retourner une réponse à l'utilisateur valide en termes de langage HTTP.



**Routing dans symfony**

Nous avons besoin d'un contrôleur "front" ou frontal : il est en charge de récupérer les informations de la requête et d'exécuter l'action correspondante qui retournera une réponse.



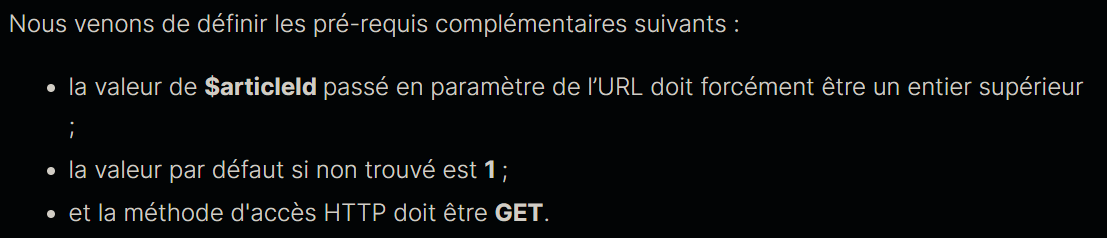
Better to use Routing !

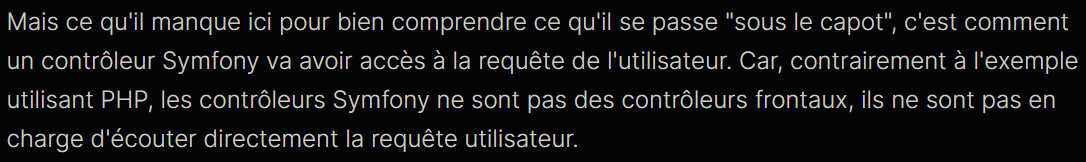
Exemple : 

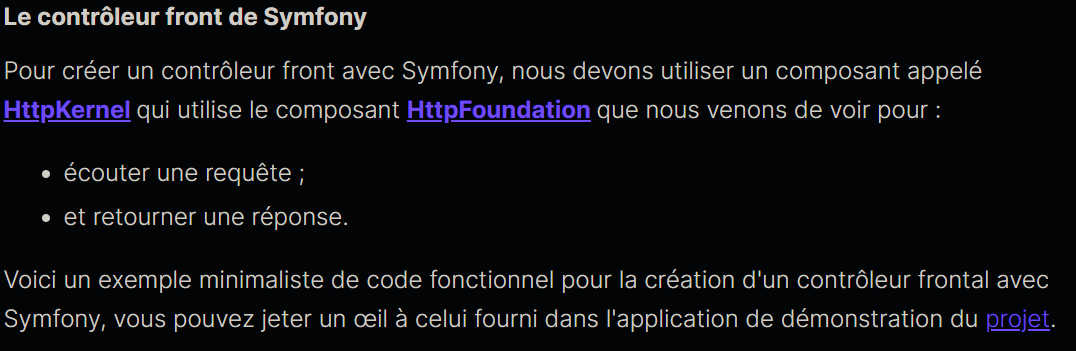
Ici, l'annotation Route s'utilise dans un bloc de commentaire particulier (il commence par 2 astérisques) et permet de définir les paramètres et contraintes de la route pour laquelle l'action (ici  home()  ou  showBlogPost($articleId)) se déclenchera.

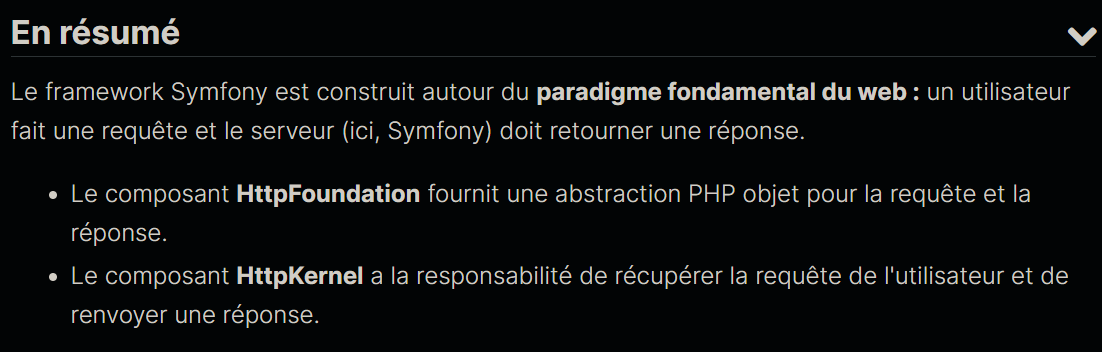
Le premier argument est la route en elle-même qui accepte des expressions régulières très puissantes !







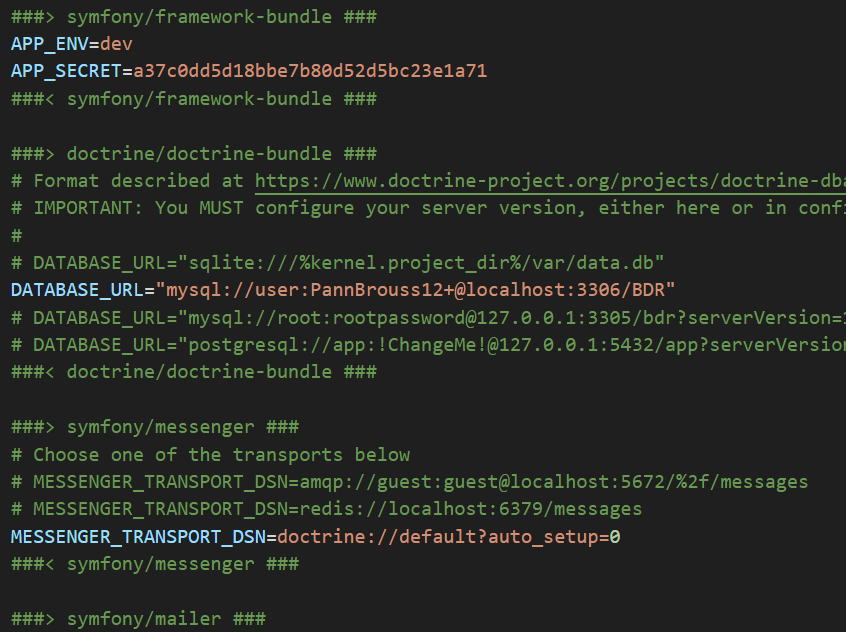




To get the long term support version.

Symfony new –webapp projetname –version=lts ;

To create a database, you must configure your env or env\_local file :



And run the command

Symfony console doctorine :database :create

Or simply

Symfony console d :d :c



And we have :



Bdr base created