SILEX

# Installation

Composer – On va pouvoir demander à un ‘composer’ de récupérer des librairies dont on a besoin, de les installer en local sur notre machine.

<https://getcomposer.org/download/> // Télécharger le composer-setup

* On laisse tout tel quel dans l’installation
* Dans un terminal, on tape ‘composer’

Silex est fourni dans une version très épurée, sans beaucoup de composants. Aussi, son créateur propose une version avec quelques composants déjà préinstallés : Silex Skeleton.

<https://github.com/silexphp/Silex-Skeleton>

* On cible le dossier choisi : cd MorganeRoulland
* Composer create-project fabpot/silex-skeleton silex\_intro
* L’installation se lance.
* <http://localhost/silex_intro/web/>
* L’application démarre.

On ouvre NetBeans IDE, on lui indique qu’on veut créer un nouveau projet avec « existing sources ». On indique le répertoire silex\_intro.

# Description de l’architecture :

**Vendor est le répertoire qui va contenir toutes les librairies.**

On ne modifie JAMAIS quelque soit le fichier présent dans VENDOR. Ce sont des librairies externes, donc on ne les touche jamais pour les garder absolument fonctionnelles. De plus, à chaque maj ou déplacement de l’application, les librairies seraient rechargées. De la même façon, c’est un répertoire qu’on ne versionne pas (= présent dans gitignore).

* Pimple est le router de Twig (moteur de template), pour fonctionner les deux vont ensemble
* On trouve aussi silex et symphony qui appelle des composants divers
* Le fichier autoload appelle des librairies externes

**Var va contenir du cache** : parce que twig lorsqu’il s’exécute retranscrit le code en php, mais pour ne pas le faire à chaque fois, il stocke les informations en cache.

**Templates :** c’est ici qu’on aura nos vues. L’éditeur reconnait les vues en twig.

**Src** : le cœur de l’application. C’est ici qu’on développe l’architecture mvc

**Config :** On observe qu’on trouve deux types de configuration si on est en développement ou en production.

**Bin :** sur symfony, on se sert fréquemment de lignes de commandes pour générer du code, mais ce n’est pas le cas sur symfony.

Dans le **répertoire web**, c’est l’entrée de notre application sur internet. On y trouve notre premier fichier index.php, on voit aussi le répertoire css.

Dans src, le fichier app crée le premier composant symfony, et lance l’application. Dans le controller, on retrouve le composant app avec un objet de la class silex $app.