INSTALLATION DE COMPOSER

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

brew install composer

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CREATION D’UN PROJET ( BLOG )

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

composer create-project symfony/website-skeleton Blog

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

INSTALLATION D’UN COMPOSANT SERVEUR (pour avoir un serveur pour lancer le projet)

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

composer require server --dev

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

LANCEMENT DU SERVEUR (prendre l’adresse http://127.0.0.1:8000)

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console server:run

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CREER UN CONTROLLER (dans notre cas son nom est : Blog)

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:controller

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DOCTRINE ET BDD\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CREER UNE BASE DE DONNEE AVEC DOCTRINE

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console doctrine:database:create

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CREER UNE ENTITE (CLASS PHP)

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:entity

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

FAIRE UNE MIGRATION DES ENTITE VERS BDD

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:migration

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console doctrine:migrations:migrate

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CREER DES FAUSSES DONNEES AVEC LES FIXTURES

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:fixtures

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CHARGER LES FIXTURES DANS LA BDD

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console doctrine:fixtures:load

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

CREER DES ENTITES

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:entity

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

METTRE A JOUR LA BASE DE DONNEE DEPUIS LES ENTITES

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console doctrine:schema:update --force

- - - - - - - - - - -- - - - - - - - - - - - - - - - - - - - - - - - -

CREER UN CONTROLLEUR

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

php bin/console make:controller

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

-> - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARBORESENCE PROJET SYMFONY\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

La liste des répertoires que contient par défaut un projet symfony 3.

**• app**

**◦ Ressources** => les templates communs, templates d’exécutions, les fichiers communs

**◦ config** => les fichiers de configurations, services, paramétrages et routes

**◦ DoctrineMigration** => Les migrations doctrine

**◦ translations** => Les fichiers de traduction

**• bin** => Les scripts exécutables

**• src** => Les bundles de notre projet

**• var** => Les logs, caches, sessions

**• vendor** => Les bundles externes

**• web** => Les fichiers publics de notre projet

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Symfony Cheat Sheet**

Useful practical commands, code and things you will frequently need for working with Symfony framework.

**Table of contents**

• Symfony Installer

• Composer

• Console

◦ General

◦ Cache

◦ Bundle

◦ Routing

◦ Doctrine

• Twig

• Controller

**Symfony Installer**

# Install Symfony Installer

curl -LsS http://symfony.com/installer > symfony.phar

sudo mv symfony.phar /usr/local/bin/symfony

chmod a+x /usr/local/bin/symfony

# Create a new project

symfony new my\_project

# Update the Symfony installer

symfony self-update

# List help

symfony about

**Composer**

# Create new project with Composer

composer create-project symfony/framework-standard-edition ~2.6

# Add package to project

composer require vendor/library

# Update all dependencies of current project

composer update

# Install dependencies for current project with versions defined in composer.lock

composer install

**Console**

**General**

# List available commands and show the Symfony version

php app/console

# Display help for given command

php app/console help [command]

# Display all configured public services

php app/console container:debug [--show-private] [service\_name]

# Dump all assets to the filesystem

php app/console assetic:dump [--watch] [--force] [--period=...] [write\_to]

# Dump the default configuration for an extension/bundle

php app/console config:dump-reference <bundle\_or\_extension\_alias>

# Send messages from spool

php app/console swiftmailer:spool:send [--message-limit=...] [--time-limit=...] [--recover-timeout=...]

# Extract translation strings from templates of a given bundle.

# It can display them or merge the new ones into the translation files

php app/console translation:update [--prefix=...] [--update-format=...] [--dump-messages] [--force] <locale> <bundle>

# Lint a template and outputs to stdout the first encountered syntax error

php app/console twig:lint <filename>`

**Cache**

# Clear the cached information

php app/console cache:clear [--no-warmup] [--no-optional-warmers]

# Warm up an empty cache

php app/console cache:warmup [--no-optional-warmers]`

**Bundle**

# Install bundles web assets under a public web directory

php app/console assets:install <target\_dir> [--symlink] [--relative]

# Generate a bundle

php app/console generate:bundle [--namespace=...] [--dir=...] [--bundle-name=...] [--format=...] [--structure]

**Routing**

# Display current routes for application

php app/console router:debug [route\_name]

# Dump all routes as Apache rewrite rules

php app/console router:dump-apache [--base-uri=...] [script\_name]

# Debug routes by simulating a path info match

php app/console router:match <path\_info>

# Display current exposed routes for an application

php app/console fos:js-routing:debug <name>

**Doctrine**

# Clear all metadata cache for an entity manager

php app/console doctrine:cache:clear-metadata [--em=...] [--flush]

# Clear all query cache for an entity manager

php app/console doctrine:cache:clear-query [--em=...] [--flush]

# Clear all result cache for an entity manager

php app/console doctrine:cache:clear-result [--em=...] [--flush]

# Create the configured databases

php app/console doctrine:database:create [--connection=...]

# Drop the configured databases

php app/console doctrine:database:drop [--connection=...] [--force]

# Convert mapping information between supported formats

php app/console doctrine:mapping:convert [--filter=...] [--force] [--from-database] [--extend=...] [--num-spaces=...] [--namespace=...] [--em=...] <to-type> <dest-path>

# Import mapping information from an existing database

php app/console doctrine:mapping:import [--em=...] [--filter=...] [--force] <bundle> <mapping-type>

# Show basic information about all mapped entities

php app/console doctrine:mapping:info [--em=...]

# Generate a new Doctrine entity inside a bundle

php app/console doctrine:generate:entity [--entity=...] [--fields=...] [--format=...] [--with-repository]

# Generate entity classes and method stubs from your mapping information

php app/console doctrine:generate:entities [--path=...] [--no-backup] <name>

# Generate a form class based on a Doctrine entity

php app/console doctrine:generate:form <entity>

# Generate a CRUD based on a Doctrine entity

php app/console doctrine:generate:crud [--entity=...] [--route-prefix=...] [--with-write] [--format=...]

# Execute arbitrary DQL directly from the command line

php app/console doctrine:query:dql [--hydrate=...] [--first-result=...] [--max-result=...] [--depth=...] [--em=...] <dql\_to\_execute>

# Execute arbitrary SQL directly from the command line

php app/console doctrine:query:sql [--depth=...] [--connection=...] <sql\_to\_execute>

# Executes (or dumps) the SQL needed to generate the database schema

php app/console doctrine:schema:create [--dump-sql] [--em=...]

# Execute (or dump) the SQL needed to drop the current database schema

php app/console doctrine:schema:drop [--dump-sql] [--force] [--full-database] [--em=...]

# Executes (or dumps) the SQL needed to update the database schema to match the current mapping metadata

php app/console doctrine:schema:update [--complete] [--dump-sql] [--force] [--em=...]

# Validate the Doctrine mapping files

php app/console doctrine:schema:validate [--em=...]

# Ensure that Doctrine is properly configured for a production environment

php app/console doctrine:ensure-production-settings [--complete] [--em=...]

# Load data fixtures to your database

php app/console doctrine:fixtures:load [--fixtures=...] [--append] [--em=...] [--purge-with-truncate]

**Twig**

{# Echo variable with html escaping #}

{{ content }}

{# Echo raw variable and disable html escaping #}

{{ content|raw }}

{# Extend template #}

{% extends '::base.html.twig' %}

**Controller**

use Symfony\Component\HttpFoundation\Response;

public function helloAction()

{

return new Response('Hello world!');

}