Les commandes Symfony Cheat Sheet

by AGD (agarciadutaitre) via cheatography.com/174798/cs/36722/

Création application : symfony new

LTS web	my_project --version=lts --webapp
Version 6.2 web	my_project --version-="6.2.*" --webapp
LTS	my_project --version=lts
La dernière version	my_project
la demo	symfony new demo –demo

LTS signifie Long Time Support soit version supportée sur le long terme.

Création inscription et connexion

	symfony console
connexion (login)	make:auth
Inscription (register)	make:registration-form
	make:user

Créer les formulaires pour l'inscription et la connexion des utilisateurs ainsi que le contrôleur user adapté

Création d'une route par configuration

Ajouter dans le fichier **config/routes.yaml**

nom_de_la_route:

path: /inscription ou / ou /machin/truc etc (url)

controller: App\Controller\nom_du_contr-oleurController::nom_de_methode

Liste des routes du projet

liste toutes les routes	php bin/console debug:router
liste une route	php bin/console debug:router <nom de la route>
liste l'enchain-ement des fonctions utilisées	php bin/console router:match <nom de la route>

Symfony

server:start --no-tls	lance le serveur en http
server:ca:install	pour un serveur en https
server:start -d	en background
server:stop	stop le serveur (ctr c)
console about	information version du projet
check:requir-ements	valide l'environnement

symfony console

list	donne la liste de toutes les commandes
liste make	donne la liste des commandes du maker
make:cont-roller --help	donne la liste des options de la commande
doctrine:dat-abase:create	Crée la db déclarée dans .env

symfony console (cont)

d:d:c	Crée la db déclarée dans .env
make:user	Crée la table user associée à l'authentification
make:e-ntity	Crée une table
make:e-ntity --rege-nerate	Généré les méthodes getter et setter manquantes
make:e-ntity --overwrite	Regénére toutes les méthodes getter et setter
make:m-igration	Génère les fichiers de migration
doctrine:-migration-s:migrate	Exécute les fichiers de migration
m:mig	Génère les fichiers de migration
d:m:m	Exécute les fichiers de migration
doctrine:-migration-s:diff	Génère un fichier de migration en comparant le schéma interne et le schéma de la base
debug:-autowiring	Liste les classes autocharg-eable

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

symfony console (cont)

debug:container	Liste les services disponibles pour l'application
dbal:run-sql 'SELECT * FROM product'	interroge la base de données

Les fichiers de migrations sont nommés migrations\Versiondateheure.php et ils contiennent les ordres sql qui vont modifier la base

php bin/console

server:dump	??
server:log	
list	liste les commandes de console
list make	liste les commandes du maker

Contrôleur

Créer un contrôleur	php bin/console make:controller nom_du_controleur
	symfony console make:controller nom_du_controleur

Les vues

Installation	composer require twig

Les vues = Twig sous symfony

Exemples de déclaration de route

```
#[Route('/api/posts/{id}', methods: ['GET', 'HEAD'])]
#[Route( '/contact', name: 'contact',
condition: "context.getMethod() in ['GET', 'HEAD'] and request.headers.get('User-Agent') matches '/firefox/i'",
// expressions can also include config parameters:
// condition: "request.headers.get('User-Agent') matches '%app.allowed_browsers%'"
)]
#[Route(
'/posts/{id}',
name: 'post_show',
// expressions can retrieve route parameter values using the "params" variable
condition: "params['id'] < 1000"
)]
// Controller (using an alias):
#[Route( '/posts/{id}', name: 'post_show',
condition: "service('route_checker').check(request)")]
use Symfony\Bundle\FrameworkBundle\Routing\Attribute\AsRoutingConditionService;
use Symfony\Component\HttpFoundation\Request;
#[AsRoutingConditionService(alias: 'route_checker')]
class RouteChecker
{
```

Exemples de déclaration de route (cont)

```
  public function check(Request $request): bool
  {
// ...
  }
}
```

php

-m	liste les modules php installés

Composer require

https://packagist.org/packages/symfony/	Liste des packages dispo sur composer
symfony/orm-pack	Installe Doctrine
--dev symfony/maker-bundle	Installe le Maker
symfony/validator	Les contraintes de validation pour Forms et DB
symfony/http-foundation	installe session
symfony/form	Installe les formulaires

Usage des annotations pour les routes

Utiliser le package annotation	composer require annotations

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Création route annotation

```
<?php
namespace App\Co ntr oller;
use Symfon y\C omp one nt \Htt -
pFo und ati on \Res ponse;
use Symfon y\C omp one nt \Rou -
tin g\A nno tat ion \Route;
class NomDuC ont rol eur Con -
troller {
   /**
     * @Route(
     * " /ch emi n_d e_u rl/ -
{pa ram 1}} ",
     * name="n om_ de_ la_ rou -
te",
     * method s={ " GET ", " -
POS T","H EAD " ,"PU T",etc},
     * requir eme nts ={"p ara -
m1": " reg exp Vou lue " },
     * defaul ts= {"pa ram -
1":"V ale urP arD efa ut"}
     * host="m y.h ost "
     * scheme s={ " htt p", " -
htt ps"}
     * priority=1
     * )
     */
    public function nomTra ite -
men t($ param1 ): Response {
         // ...

return $this- >re nde r(' nom -
_re p_t emp lat e/n om_ du_ tem -
pla te.h tm l.t wig', [
                        'no -
mDu Par ametre' => $valeu r_d -
u_p ara metre,
'param1' $param1,
                ]);
   }
}
```

Option pour les routes

name="routeName"	Donner un nom à une Route
methods={"method1", "method2"}	retreindre l'accès au requête HTTP de type
Type de requête HTTP	GET,POST,-DEL,PUT,HEAD etc
host="nom ou IP du host"	retreindre l'accès à un host en particulier
requirements={"param1":"regexp1", "param2":"regexp2"}	appliquer des règles aux paramètres
defaults={"param1":"default1", "param2":"default2"}	Valeur par defaults des paramètres

Création d'un contrôleur en lien avec la base

```
//
src/Controller/ProductController.php
namespace App\Co ntr oller;
// ...
use App\En tit y\P roduct;
use Doctri ne \Per sis ten ce -
\Man age rRe gistry;
use Symfon y\C omp one nt \Htt pFo -
und ati on \Res ponse;
use Symfon y\C omp one nt \Rou tin -
g\A nno tat ion \Route;
class Produc tCo ntr oller extends
Abstra ctC ont roller
{
        #[R out e(' /pr oduct',
name: 'creat e_p rod uct')]
```

Création d'un contrôleur en lien avec la base (cont)

```
>     public function createProduct(Mana-
gerRegistry $doctrine): Response
    {
        $entityManager = $doctrine->getMana-
ger();
        $product = new Product();
        $product->setName('Keyboard');
        $product->setPrice(1999);
        $product->setDescription('Ergonomic
and stylish!');
        // tell Doctrine you want to (eventually)
save the Product (no queries yet)
        $entityManager->persist($product);
        // actually executes the queries (i.e. the
INSERT query)
        $entityManager->flush();
        return new Response('Saved new
product with id '.$product->getId());
    }
}
```

Tester le contrôleur avec : http://localhost:8-000/product
Vérifier la base de donnée : php bin/console dbal:run-sql 'SELECT * FROM product'
https://symfony.com/doc/current/doctrine.h-tml#migrations-creating-the-database-table-s-schema

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com