DOCUMENTATION SUR SYMFONY TWIG

1 – Notion de Framework

Framework est formé des mots « *frame* » et « *work* » et signifie donc « cadre de travail ». Concrètement, il s'agit d'un ensemble de composants servant à créer les fondations, l'architecture et les grandes lignes d'un logiciel. Il a pour but d'améliorer la productivité des développeurs qui l'utilisent.

a – Etude comparative des différents Framework PHP

Il existe plusieurs Frameworks php mais cette étude portera sur les plus utilisés :

CakePHP

CakePHP est relativement simple à prendre en main comparé à certains autres frameworks PHP tels que Zend, pour ne citer que l'un des plus connus. Il permet le développement rapide d'applications (RAD : Rapid Application Development) et simplifie leur maintenance. Les principes de base de CakePHP sont avant tout sa robustesse et sa simplicité de mise en œuvre quel que soit le type d'application. Il la rend facilement extensible et personnalisable avec l'architecture MVC.

Les principales caractéristiques de CakePHP sont :

- Compatible PHP 4 et PHP 5.
- Pas de configuration nécessaire.
- Utilisation d'une simplicité peu commune.
- Communauté conséquente et active.
- Licence MIT (toute application développée peut être vendue).
- Entièrement conçu et maintenu par l'équipe de développeurs.
- Intègre l'architecture MVC.
- Gestion facilitée des bases de données grâce à Active Record.
- Entièrement Orienté Objet (OO).
- Nombreux outils de développement (ajax, formulaires…).
- Génération d'applications CRUD (Create/Read/Update/Delete)

Son architecture extensible permet de développer et maintenir les applications en optimisant les tâches du développeur et favorisant un gain de productivité non négligeable. Ces avantages sont rendus possibles grâce à l'adoption des systèmes MVC (Modèle-Vue-Contôleur) et ORM (Object-Relational Mapping).
Framework d'applications rapides (RAD) CakePHP est avant tout de

conception robuste. Il prend en charge tous les aspects d'une page web, de la requête initiale à l'interface finale. En appliquant les principes de l'architecture MVC, il permet toute forme de personnalisation et d'extension de chaque application. Reposant sur un concept simple et puissant, une organisation structurée, CakePHP se positionne parmi les meilleurs frameworks et offre aux développeurs toutes les facilités nécessaires à l'élaboration rapide d'applications sérieuses et complexes en leur assurant consistance et logique.

CodeIgniter

CodeIgniter est un framework PHP puissant, bien que léger, basé sur le fonctionnement de Ruby on Rails et adoptant l'architecture MVC (Modèle-Vue-Contrôleur). Développé à l'origine pour les propres besoins de l'entreprise EllisLab en 2006, il a depuis adapté le statut open source. CodeIgniter est particulièrement simple à utiliser, performant et d'une vitesse d'exécution remarquable. Il est fortement recommandé aux développeurs débutants.

Les principales caractéristiques de CodeIgniter :

- Compatible PHP 4 et PHP 5
- Simplicité de prise en main
- Intégration de l'architecture MVC
- Nombreuses librairies
- Complètement orienté objet
- Importante communauté très active (anglophone en majorité)
- Abondante documentation
- Souplesse et évolution constante

CodeIgniter est un framework PHP qui comme tout framework permet de réduire le temps de développement et d'optimiser le code d'une application.
Il est particulièrement souple et adapté aux débutants car aisément compréhensible. Moins connu que CakePHP ou Zend, CodeIgniter est maintenu par la Société EllisLab et remarquablement documenté. Son avenir semble prometteur, à en juger par l'importance de sa communauté. CodeIgniter est puissant bien que nécessitant peu de ressources. Il est en constante évolution et son importante collection de librairies contribue largement aux gains de productivité que l'on attend d'un framework, en matière de développement d'applications de toutes natures.. Sa simplicité d'utilisation par rapport à Symfony ou Zend contribue à sa popularité au sein des développeurs PHP.
CodeIgniter est le framework idéal pour le développement de petites et moyennes applications. En effet il nécessite peu de ressources tout en

offrant un maximum de fonctionnalités. Son système de cache lui procure d'excellentes performances et minimise l'impact sur le chargement des pages.

Ses principaux atouts : extrême légèreté, robustesse et fiabilité. Facile à installer et acceptant de multiples environnements, CodeIgniter à tout pour séduire.

Selon un test réalisé il y a quelques mois par Richard Goutorbe, consultant informatique à Montpellier, l'utilisation de CodeIgniter pour le développement d'applications PHP offre un gain de productivité de 50%. La perte de performance liée à son exécution est de l'ordre de 10%, très largement compensée par une augmentation de fiabilité et de maintenabilité due à la réutilisation du code.

Yii

Le Yii Framework ("Yes, It Is") est un framework pour PHP 5 et utilise le paradigme de programmation orientée objet. Il est destiné au développement d'applications Web. Yii nécessite minimalement la version 5.1.0 de PHP. Une documentation complète est disponible. La communauté autour du projet est très active. Le créateur et développeur principal de Yii est Qiang Xu, qui a également développé et maintenu le framework PHP Prado pendant 3 ans. Yii est d'ailleurs le successeur officiel de Prado.

Il offre notamment les fonctionnalités suivantes :

- de hautes performances ;
- une architecture Modèle-Vue-Contrôleur ;
- un accès aux bases de données par DAO/ActiveRecord ;
- les fonctions nécessaires pour la gestion de l'internationalisation (I18N/L10N) ;
- la gestion de caches ;
- le support de AJAX via l'intégration de jQuery ;
- le contrôle d'accès par la gestion de rôles utilisateurs (RBAC) ;
- la génération automatique du code PHP pour les opérations de base (création, lecture, mise à jour et suppression) sur la base de données (scaffolding) ;
- le contrôle des saisies utilisateurs sur les formulaires ;
- la notion de widgets ;
- les événements sur les éléments des pages (boutons, liens...) ;
- la gestion de thèmes pour l'habillage des sites ;
- le support des services Web ;

- la possibilité d'ajouter des fonctions via un système de plugins ;
- le support des tests unitaires et fonctionnels.
- la migration de bases de données ;

Son architecture extensible permet de développer et maintenir les applications en optimisant les tâches du développeur et favorisant un gain de productivité non négligeable. Ces avantages sont rendus possibles grâce à l'adoption des systèmes MVC (Modèle-Vue-Contrôleur) et ORM (Object-Relational Mapping).

Symfony

Symfony est un framework open source écrit en PHP 5, utilisant l'architecture MVC.
Conçu dès l'origine en 2005 par une société française, Sensio Labs, bien connue pour ses idées novatrices dans le développement web, Symfony est robuste, fiable, très puissant, et s'adapte à un très grand nombre de configurations.
Développé par une équipe de professionnels expérimentés, il est spécialement dédié à la conception d'applications moyennes à très lourdes, c'est pourquoi il trouve tout naturellement sa place dans l'entreprise. Ses multiples fonctionnalités et son abondante documentation ont fait de Symfony, en quelques années, l'un des frameworks PHP les plus utilisés.

Les principales caractéristiques de Symfony :

- Compatible PHP 5 totalement orienté objet.
- Intègre l'architecture MVC et la méthode ORM.
- Licence MIT (tout ce qui est réalisé avec Symfony peut être vendu).
- Extensible et modulaire.
- Supporte ajax.
- Système de templates.
- Importante communauté très active.
- Très grand nombre de traductions.
- Conçu pour une utilisation professionnelle.

Comme chaque framework le but de Symfony est d'abord de faciliter et d'accélérer le temps de développement par sa puissance, tout en optimisant le code en en permettant sa réutilisation, en toute sécurité. Un des nombreux avantages de Symfony est sa facilité d'installation sur la plupart des configurations existantes. Sa compatibilité avec la quasi-totalité des bases de données fait de Symfony un framework que les développeurs web confirmés ont peu de difficultés à prendre en main rapidement. Bien plus

qu’un framework MVC, Symfony réunit les meilleurs outils de développement PHP afin d’aborder avec cohérence la réalisation d’applications web. L’architecture, outils et composants de Symfony favorisent et facilitent la conception d’applications complexes avec une aisance et une rapidité certaine.

Basé sur l’expérience, fiable et multi-fonctionnel, Symfony gagne de jour en jour en notoriété et reconnaissance. Les nouveaux arrivants trouvent auprès de l’importante communauté, appui, documentation et contributions. En effet, le développement actuel de Symfony repose pour une bonne partie sur l’ajout de modules ou plugins (plus de 600 aujourd’hui), très faciles à produire et utiliser. Fabien Potencier, créateur et responsable du développement de Symfony et PDG de Sensio Labs, confirme l’adoption de cette méthode dans les évolutions futures. Il désigne Zend comme son principal concurrent.

Laravel

Laravel a été créé par Taylor Otwell1 en juin 20112. Le référentiel Larave/laravel présent sur le site GitHub contient le code source des premières versions de Larave. À partir de la quatrième version, le framework est développé au sein du référentiel Laravl/framework. En peu de temps, une communauté d'utilisateurs du framework s'est constituée. La quatrième version de Laravel est écrite en PHP 5.3.73 et son installation est basée sur le gestionnaire de paquets Composer. Dans une syntaxe élégante, Laravel fournit des fonctionnalités en termes de routage de requête, de mapping objet-relationnel (un système baptisé Eloquent implémentant Active Record), d'authentification, de migration de base de données, de gestion des exceptions et de test unitaire1. L'ensemble des éléments constitué de la documentation du framework, de tutoriels et de copies d'écran permettent l'apprentissage du framework.

Zend

Zend Framework est un framework PHP 5 créé par la société Zend Technologies, appelée aussi The PHP Company car fondée par les pères du PHP moderne : Andi Gutmans et Zeev Suraski. Il est distribué open source sous la New BSD license. De par son histoire, Zend tient une place particulière dans l’univers des Frameworks PHP. Bénéficiant d’une communauté massive à travers le monde, il est aujourd’hui à n’en pas douter le plus connu et utilisé des frameworks. Notons que son utilisation requiert toutefois une connaissance approfondie de PHP. Zend

Technologies l'a développé en privilégiant les « bonnes pratiques ». Il est totalement orienté objet et intègre l'architecture MVC.

Les principales caractéristiques de Zend Framework :

- Compatible PHP 5
- Conception totalement orientée objet
- Intègration de l'architecture MVC (Modèle-Vue-Contrôleur)
- Faible dépendance des composants entre eux, permet leur utilisation individuelle (use-at-will)
- Puissant, extensible et modulaire
- Imposante communauté hyper-active
- Fortement adapté à l'environnement professionnel
- Développeurs externes respectant la licence open source
- Alliés et contributeurs hors normes : Microsoft, Google…

Zend Framework peut être considéré actuellement comme le leader des Frameworks PHP. Sa souplesse est telle, qu'à partir d'une base quelconque, il est possible de créer une liaison avec d'autres frameworks ou composants.
D'une puissance, robustesse et fiabilité à toute épreuve, la conception de Zend Framework est particulière et le différencie des autres frameworks PHP. En effet, ses composants ont la faculté d'être utilisés individuellement, car très peu dépendants des uns envers les autres. Par contre, utilisés ensemble dans la librairie standard de Zend Framework, ils donnent au framework sa redoutable puissance. De même, bien que l'architecture MVC soit présente nativement dans Zend Framework, elle n'est pas imposée mais optionnelle. Toujours à la pointe du développement, de multiples composants permettent l'accès aux services web les plus modernes. Le principal moteur du projet reste évidemment Zend Technologies, mais certaines sociétés extérieures ont développé leur propres composants et les ont mis ensuite à la disposition de la communauté. Parmi ces contributeurs exceptionnels, ils convient de citer Google et Microsoft, deux des plus importants acteurs du web. Zend Framework est en perpétuel développement, et chaque entreprise intéressée peut développer ses composants, à la seule condition de les mettre gracieusement à la disposition des utilisateurs, dans le respect de la licence. Le concept PHP reste l'esprit de Zend Framework : libre, ouvert, simple, efficace et puissant !!!

Nom	PHP	Modèle	ORM	Template	Cache	Url conviviales	Validation de formulaire	Ajax	Extension	Générateur de code
Cake PHP	4/5	MVC	AR	PHP	0	0	0	0	0	0
CodeIgniter	4/5	MVC	AR	PHP	0	0	0	N	0	N
Yii	5	MVC	DAO	Twig ou Smarty	0	0	0	0	0	0
Laravel	5	MVC	AR	PHP	0	0	N	N	0	N
Symfony	5	MVC	Propel	PHP ou smarty	0	0	0	0	0	0
Zend	5	MVC	/	PHP	0	0	N	N	N	N

Légende
0 = oui
N= non
AR= Active Record
MVC = Modèle-Vue-Contrôleur
Propel= Mapping objet relationnel (ORM)
DAO= Data Access Object

b – Historique de symfony

La première version de Symfony sort le 18 octobre 2005. Elle est développé par SensioLabs à l'époque une simple agence web qui à force de recréer les mêmes fonctionnalités de gestion d'utilisateurs, gestion ORM, etc. va décider de développer un framework pour ses propres besoins. Le projet prend alors le nom de Symfony suite à la volonté de Fabien Potencier (le créateur du framework) de conserver les initiales S et F de Sensio Framework. Dans cette première version, le framework compatible à partir de la version de PHP 5.2.4 embarque les fonctionnalités suivante :

- Une séparation en trois couches selon le modèle MVC ce qui permet une meilleure maintenabilité et évolutivité.

- Des performances optimisées et un système de cache afin d'assurer des temps de réponse optimaux.

- Une gestion des URL parlante, permettant à une page d'avoir une URL distincte de sa position dans l'arborescence.

- Un système de configuration en cascade utilisant le langage YAML.

- Un générateur de back-office et un lanceur de module.

- L'internationalisation native.

- Une couche de mapping objet-relationnel (ORM) et une abstraction de données.

- Le support d'AJAX.

- Une architecture extensible permettant la créations et l'utilisations de plugins.

La version 2 de Symfony casse la compatibilité avec la branche 1.x. et est disponible à partir de PHP 5.3.3. Sortie le 28 juillet 2011. En effet, la deuxième mouture du framework vient par défaut avec une boîte à outils impressionnante pour permettre aux développeurs d'être immédiatement opérationnel. Le support pour des librairies telles qu'Assetic, Twig, Imagine et Monolog est une illustration de la volonté de pouvoir réutiliser tout ce qui se fait de mieux en termes de performances et de standards. Le concept de « Bundle » commence à arriver, d'ailleurs Symfony2 est un Bundle.

La troisième version de Symfony arrive le 30 novembre 2015. Elle est développé pour fonctionné avec une version minimum de PHP en 5.5.9. Si l'arrivé de Symfony 2 avait soufflé un vent de panique dans la communauté avec la difficulté de porter un projet Symfony 1 sur Symfony 2, la troizième version permet une mise à niveau plus sereine. La bonne nouvelle est donc que le code Symfony 2 est compatible avec Symfony 3.

Symfony 4 est sorti le 30 novembre 2017. Elle représente une refonte complète de ses idées et fonctionnalités pour correspondre aux pratiques de l'industrie: les bundles d'application ont disparu, les paramètres de configuration sont désormais des variables d'environnement, la structure du répertoire de l'application est plus facile à parcourir. Le résultat est moins de concepts Symfony et plus de pratiques standard. Plus facile à apprendre, plus facile à configurer, plus facile à installer et à déployer et plus facile à maîtriser.

Symfony 5 est sorti le 21 novembre 2019 et représente la version la plus récente de Symfony à ce jour. Elle poursuit la révolution entamée par Symfony 4 pour créer la meilleure version Symfony jamais publiée. Encore plus facile à apprendre, plus facile à configurer et plus facile à installer, à déployer et à maîtriser. Symfony 5 s'intègre de manière transparente à Symfony Flex pour automatiser les tâches les plus courantes effectuées par

les applications. L'activation des bundles ou la création de leur configuration initiale est oubliée: Symfony Flex est là.

2 – Composer

a – Quoi ?

Composer est un outil permettant de gérer les dépendances en PHP. Une dépendance est une bibliothèque dont votre projet dépend pour fonctionner. Par exemple si un projet utilise la bibliothèque SwiftMail pour envoyer des e-mails alors SwiftMail est une dépendance dans ce projet.

b – Avantages et inconvénients

Composer est utile dans le sens où il télécharge automatiquement les librairies dont un projet a besoin pour fonctionner, il dispose également d'un autoloader permettant automatiquement de charger tout ce dont vous avez besoin et tout ce qu'il y a à faire est d'inclure un fichier. Il permet également d'utiliser les namespace psr-4 pour charger un chemin spécifique sur le projet et le faire inclure dans le fichier du chargeur automatique, ce namespace pourra simplement être utilisé et sera ensuite disponible dans toute l'application.

c – Installation

Composer étant un outil PHP, il faut vérifier l'environnement de PHP et vérifier qu'il soit bien configuré pour pouvoir s'exécuter via la console.

Sous Windows il faut lancer l'invite de commande et taper la commande
php -v

Sous Linux et Mac lancer le terminal et exécuter la même commande

Si cette commande retourne la version de PHP et d'autres informations, c'est tout bon pour vous. Il faut aussi vérifier que la version de PHP est 7.1.3 au minimum sinon il faut d'abord le mettre à jour.

En cas d'erreur c'est-à-dire que Windows ne sait pas où trouver le dossier PHP, voici la démarche à suive :

1. Aller dans les paramètres système avancés :
   Démarrer > *Panneau de configuration* > *Système et sécurité* > *Système* > *Paramètres système avancés* ;
2. Cliquer sur le bouton `Variables d'environnement…` ;
3. Dans la section `Variables utilisateurs` trouver l'entrée `Path` et la sélectionner ;
4. Cliquez sur `Modifier` puis sur `Nouveau` puis entrer votre répertoire PHP à la fin, c'est le répertoire dans lequel se trouve le fichier `php.exe`. Par exemple sur **WAMP** :
   `C:\wamp\bin\php\php7.4.0` ou sous **XAMPP** : `C:\xampp\php`
5. Confirmez en cliquant sur OK. Vous devez ensuite redémarrer l'invite de commandes pour prendre en compte les changements.

Si vous êtes sous Linux, vérifiez votre installation de PHP. Vous devez notamment avoir le paquet php7-cli, qui est la version console de PHP.

Passons à l'installation de Composer : si vous êtes sous Windows, la méthode la plus facile est d'installer Composer via son exécutable d'installation, comme n'importe quel autre logiciel c'est-à-dire depuis un lien comme celui-ci par exemple : getcomposer.org/Composer-Setup.exe puis suivre les instructions.

Sous Linux et Mac exécuter simplement la commande suivante :
php -r "eval('?>'.file_get_contents('http://getcomposer.org/installer'));"

Si tout est ok le terminal devrait afficher :
All settings correct for using Composer
Downloading...

Composer (version 1.6.3) successfully installed to:
C:\wamp\www\admin\composer.phar
Use it: php composer.phar

Cependant pour récupérer certaines bibliothèques Composer utilise git qu'il faudra installer aussi.

2) Symfony

Symfony est un framework php

a) Installation

Les pre-requis pour installer symfony

- une version de php 7.2.5 ou supérieur.
- Installer Composer est utiliser pour installer les packages php

La création d'un nouveau projet se fait en ligne de commande et peux se faire en deux façons :

1- first

```
symfony new my_project_name --full
```

2- second

```
symfony new my_project_name
```

Pour ouvrir le projet on utilise la commande

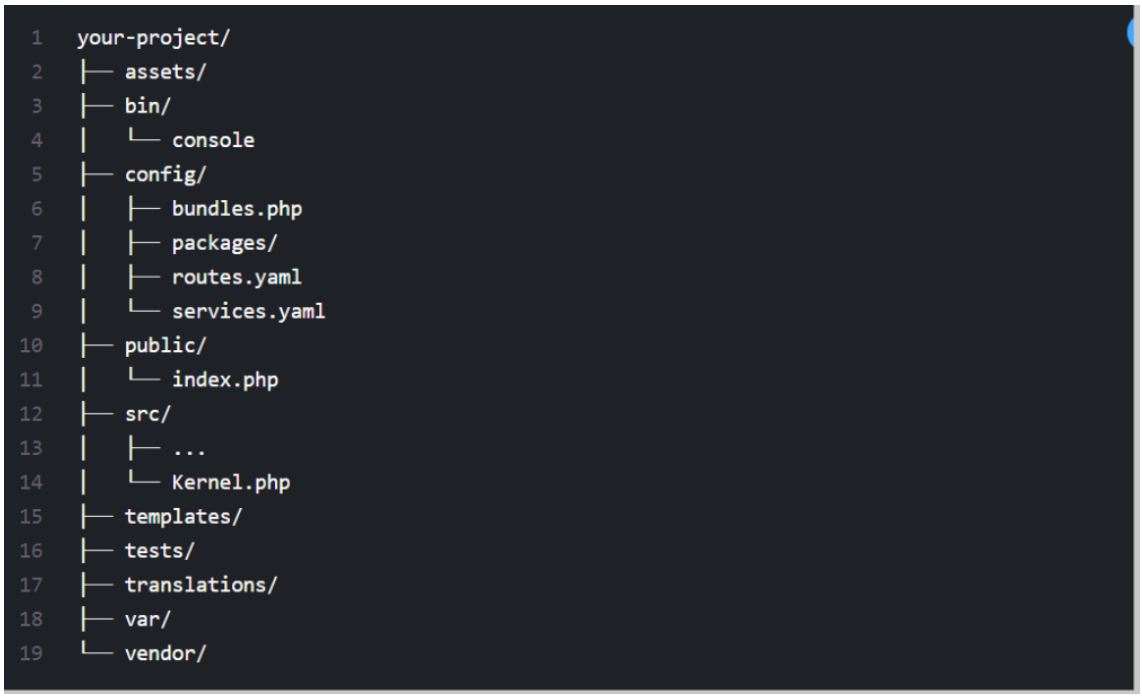
```
symfony server:start
```

● Fichier yml(Rôle et Structure)

Les fichiers YML contiennent du code source écrit dans le langage de programmation YAML (YAML Ain't Markup Language) et sont utilisés par les développeurs. Le code contenu dans un fichier YML est lisible par l'homme et généralement utilisé pour sérialiser des données. Un fichier YML permet de lire et d'écrire des données indépendamment du langage de programmation. Ainsi, les fichiers YML peuvent être utilisés avec de nombreux autres fichiers contenant du code source écrit via différents langages tels que C, C#, C++, Java, PHP, etc.

● Structure d'un projet symfony(rôle de chaque dossier)

La structure d'un projet symfony



Config/ : contient la configuration

Src/ : contient tout le code php

Template : contient toutes les modèles de twig

Bin/ :

Var/ : c'est le cache

Vendor/ : contient les bibliothèques tierces qui sont telecharge via le gestionnaire des packages.

Public/ : c'est la racine du document de notre projet . on y met tous les fichiers accessible au public.

b) Symfony Flex : Notion de bundle

Un Bundle est un répertoire dans un projet symfony qui intègre une structure bien définie, ce répertoire permet d'implémenter plusieurs fonctionnalités qui peuvent être utilisées dans d'autre projet symfony. On peut voir un Bundle comme un plug-in dans symfony. Ce framework propose des milliers de Bundles réutilisables quasiment dans tous les projets symfony qu'on peut avoir. il existe même des sites qui proposent de télécharger des Bundles, il y'a par exemple : KnpBundle. Pour créer un bundle on peut procéder de trois manières :

c) Via Composer on utilisant cette commande
d) $ composer.phar require nomdubundle/exemple-bundle "version", en ligne de commande ou manuellement.
e) Utiliser un Bundle peut apporter plusieurs avantages et surtout faire gagner beaucoup de temps, ça permet aussi de travailler sur une base solide, car la plupart des Bundles existants ont été construits pour être flexible et réutilisable en plus d'être sécurisé.

f)Architecture d'un bundle

g) Un bundle a généralement toujours la même structure, image ci-dessous, mais ne se limite pas qu'au dossier indiqué on peut aussi rajouter des dossiers dans le bundle afin d'organiser le projet plus proprement et de faire en sorte que chaque partie du dossier soit dédier aune tache particulière.

```
ExempleBundle
        Command/
        Controller/
        DependencyInjection/
        Entity/
        Resources/
                config/
                public/
                translation/
                views/
        Tests/
```

h)
i)
j)
k)
l)
m)
n)
o)
p) Eléments d'un Bundle
q) **Command** : dans ce répertoire, on aura les classes qui vont permettre de construire des commandes du Bundle qui vont être utilisées dans le terminal dans le projet.
r)**Controller** : c'est ici qu'il aura des classes et fonctions appelées controlleur. Le rôle d'un contrôleur est de créer et de retourner une réponse aux vues de l'application
s) **DependanceInjection** : dans ce dossier, il y'a des méthodes qui vont permettre de gérer la dépendance d'injection. Ceci permet surtout de rendre utilisable nos services symfony dans toute l'application et de laisser à symfony de s'occuper des taches répétitives qu'on pourra faire notamment instancier des classes.
t)**Resources**

u) - **config** : Contiens tous les fichiers de configuration du Bundle (service, routing)
v) - **doc** : éventuelles documentations du bundle
w) - **translation** : Ce dossier va contenir les différents fichiers de traduction du bundle si l'application est développée en plusieurs langues.
x) - **public** : Ce répertoire contient tous les astes du Bundle (css, js, images etc.)
y) - **views** toutes les vues qui vont être retournées par le contrôller.
z) **Entity** : le modèle de l'application il y'aura des classes qui vont contenir des attributs et leurs getters/setters pour gérer la couche métier du Bundle ainsi de faire correspondre chaque attribut de la classe par un champ de la table de la base de données tout ceci est géré par doctrine.
aa) **Tests** : Dossier qui contient tous les tests de l'application.
bb)

c) Commandes de Génération

- Controller

Pour créer un controller symfony nous la tache avec une generation en ligne de commande

```
php bin/console make:controller BrandNewController

created: src/Controller/BrandNewController.php
created: templates/brandnew/index.html.twig
```

- Entity et des Relations

```
$ php bin/console make:entity

Class name of the entity to create or update:
> Product

New property name (press <return> to stop adding fields):
> name

Field type (enter ? to see all types) [string]:
> string

Field length [255]:
> 255

Can this field be null in the database (nullable) (yes/no) [no]:
> no
```

- Maker

d)Routing(yml, annotation)

les rpotes peuvent être configure en php, yaml, xml ou en utilisant les annotations. Pour créer une route en tant qu'annotation on peut exécuter la commande :

```
> composer require annotations
```

Cette commande installe les dependances et cree un fichier de configuration qui indique a symfony de rechercher toutes les routes defini comme annotation dans n'importe quelle classe php defini dans le dossier controller.

e) ORM: Doctrine

Doctrine est le meilleur ensemble de bibliothèque php qui fournit tous les outils nécessaire pour travailler avec la base de donnée en symfony. Ces outils prennent en charge les bases de données relationnelles(SGBDR) comme MYSQL et PostgreSQL.
Pour utiliser doctrine il nous faut configurer la base de donnee dans le fichier avec l'extension .env.

```
# .env (or override DATABASE_URL in .env.local to avoid committing your changes)

# customize this line!
DATABASE_URL="mysql://db_user:db_password@127.0.0.1:3306/db_name"

# to use sqlite:
# DATABASE_URL="sqlite:///%kernel.project_dir%/var/app.db"
```

DATABASE_URL contient les informations de notre connexion a la base de donnée.

- Génération de la Base Donnée

Pour générer la base de donnée on exécute la commande suivante :

```
> php bin/console doctrine:database:create
```

- Les Migrations

La migration permet de convertir les classes en table dans la base de donnée en exécutant les commandes suivante.

```
> php bin/console make:migration
```

```
> php bin/console doctrine:migrations:migrate
```

- Entity Manager(persist,remove,....)

Pour faire persister les données dans la base de donnée il faut avoir un controller du même nom que l'entité.

```php
class ProductController extends AbstractController
{
    /**
     * @Route("/product", name="create_product")
     */
    public function createProduct(): Response
    {
        // you can fetch the EntityManager via $this->getDoctrine()
        // or you can add an argument to the action: createProduct(EntityManagerInterface $entit
        $entityManager = $this->getDoctrine()->getManager();

        $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();
```

- Les Repository et DQL(Doctrine Query Language)