



Ressourcesinformatiques



+ QUIZ

Version en ligne

OFFERTE !

pendant 1 an

Apprendre à développer des applications web avec **PHP** et **Symfony**

En téléchargement



code source des exemples

Yves ROCAMORA





Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence ENI de l'ouvrage **RIPHSYM** dans la zone de recherche
et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1
Qu'est-ce qu'un bon développeur ?

- 1. Développer des applications web avec PHP et Symfony 11

Chapitre 2
La boîte à outils

- 1. Windows ou Linux ? 15
 - 1.1 Windows 16
 - 1.2 Linux 16
 - 1.3 macOS 17
- 2. La relation client-serveur 18
- 3. Le serveur local 20
- 4. Visual Studio Code 25

Chapitre 3
Le langage PHP

- 1. Comment écrire du PHP ? 31
- 2. Les bases du langage : votre premier Hello World ! 33
- 3. Les variables en PHP 35
- 4. Les structures de contrôle 38
 - 4.1 La structure de contrôle if 38

2 _____ Apprendre à développer

des applications web avec PHP et Symfony

4.2	Variable booléenne	43
4.3	Structure de contrôle foreach	44
4.4	Opérateur ternaire	48
5.	Les fonctions PHP	49
5.1	Paramètres d'une fonction	51
5.2	Valeurs par défaut des paramètres	54
5.3	Retour de fonction	55

Chapitre 3

Le langage Objet

1.	Introduction	59
2.	Des objets en programmation	59
3.	Les classes	60
4.	Les objets	62
5.	Les propriétés et les méthodes d'un objet	65
6.	Pourquoi le langage est-il dit objet ?	66
7.	L'objet \$this	66
8.	Les méthodes magiques	72
8.1	La méthode __toString()	72
8.2	Les méthodes __get () et __set() : portée des éléments	74
8.3	Les méthodes __construct et __destruct	81
9.	Les espaces de noms	85
9.1	Un espace de noms, pour quoi faire ?	88
10.	L'héritage de classe	94
10.1	Classe mère et classes filles	96
11.	La signature d'une méthode	97
12.	Redéfinition d'une méthode	99
13.	Portée des éléments dans les classes filles	101
14.	Redéfinition du constructeur de la classe mère	102

15. Les constantes et les variables « static »	107
15.1 Les constantes	107
15.2 Les variables « static »	109
16. Les classes abstraites et les interfaces	112
16.1 Les classes abstraites	112
16.2 Les interfaces	117
17. Conclusion	119

Chapitre 5 Les frameworks

1. Un framework : pour quoi faire ?	121
1.1 Quels sont les avantages ?	121
1.2 Quels sont les inconvénients ?	122
2. Les frameworks PHP	123
3. Le framework Symfony	124
3.1 Présentation du framework	124
3.2 La documentation	126

Chapitre 6 L'outil Composer

1. Introduction	129
2. Variables d'environnement	132
3. Installation de Composer	136
4. Utilisation de Composer	137

4 _____ Apprendre à développer

des applications web avec PHP et Symfony

Chapitre 7

Installation de Symfony

1. La bonne configuration 147
2. L'installation de l'installateur ! 148
3. L'installation d'un projet 149
4. L'installation de Symfony avec Composer 150
5. L'installation d'une application Symfony allégée 150
6. Le test de l'installation 150
7. Le serveur local de Symfony 151

Chapitre 8

Configurer une application

1. Le fichier .env 153
 - 1.1 Liste des variables d'environnement 156
 - 1.2 Remplacement des variables d'environnement en local 157
 - 1.3 La création d'une nouvelle configuration 157

Chapitre 9

Une première application

1. La structure de Symfony 159
2. Les contrôleurs 160
3. Les vues 163
4. Le dossier public 166
5. Le dossier var. 166
6. Le dossier vendor 167
7. Les autres fichiers de l'application 167
8. Les composants de HttpFoundation 168
9. L'objet Request 169

10. L'objet Response	175
11. Les variables de session.	182
11.1 L'utilité des variables de session.	182
11.2 L'utilisation des variables de session sous Symfony	183
11.3 Les Flash Bags	185

Chapitre 10 Le routage

1. Organisation de l'application.	189
2. Intérêt des routes	190
3. Routes sans annotations	191
4. Verbes des routes	192
5. Paramètres des routes	193
6. Paramètres conditionnels	194
7. Validation des paramètres	195
8. Liste des routes	197

Chapitre 11 Le moteur de template Twig

1. La syntaxe	199
2. L'héritage.	202
3. L'inclusion de vue	205
4. L'utilisation des variables d'environnement	207
5. Les sessions et les Flash Bags dans Twig	208
5.1 Les variables de session.	208
5.2 Les Flash Bags	208
6. L'inclusion du CSS et du JavaScript dans une vue	211
7. L'utilisation des routes dans la vue	213

6 — Apprendre à développer

des applications web avec PHP et Symfony

8. Les filtres et les fonctions.	215
8.1 Les filtres.	215
8.2 Les fonctions.	219

Chapitre 12

Webpack Encore

1. Introduction	221
2. Utilisation de Sass	224
3. Utilisation de Vue.js.	227
4. Utilisation de app.css et app.js dans les vues	228

Chapitre 13

Le profiler de Symfony	233
----------------------------------	-----

Chapitre 14

Symfony Flex	235
------------------------	-----

Chapitre 15

La couche modèle avec Doctrine

1. Introduction	239
2. Les bases de données.	239
3. Le langage SQL	240
4. L'ORM de Symfony : Doctrine	241
5. Les entités	243
6. Les migrations.	248
7. Les fixtures	251
8. La récupération des données à partir de la base	256

9. Les méthodes du Repository	267
10. Le langage DQL	269
11. Le Query Builder	271
12. L'exécution des requêtes SQL	274
13. L'écriture d'une requête SQL et l'obtention des objets mappés . . .	275
14. Les relations entre entités	276
15. Les relations OneToOne	278
16. Les relations ManyToMany	284
17. Les relations bidirectionnelles	290
18. Les relations bidirectionnelles avec attributs	294
19. Le Lazy Loading	295
20. Le Reverse Engineering	299

Chapitre 16

Les formulaires

1. Introduction	301
2. Form Builder	302
3. Formulaires externalisés	308
3.1 Définition	308
3.2 Utilisation du formulaire externe	312
4. Personnalisation de l'affichage d'un formulaire	313
5. Traitement des données du formulaire	319
6. Récupération des données de l'entité par défaut	325
7. Ajout des boutons de mise à jour dans la vue liste	327
8. Suppression d'une entité	331
9. Traitement de la jointure OneToOne	331
10. Traitement de la jointure ManyToMany	334

8 _____ Apprendre à développer

des applications web avec PHP et Symfony

11. Type EntityType	343
12. Création de types de champ personnalisés	345
13. Validation des formulaires	349
13.1 Règles de validation	349
13.2 Service Validator	353
13.3 Asserts sur un accesseur	354
13.4 Contraintes de validation sur un callBack	356
13.5 Contraintes de validation sur la classe	359
13.6 Registration group	361
13.7 Création de ses propres contraintes	363
13.8 Personnalisation des messages d'erreurs : les fragments	367

Chapitre 17

La sécurité

1. Introduction	373
2. L'authentification	374
3. L'autorisation	384
3.1 access_control	385
3.2 Accès contrôleur	389
3.3 Accès action	390
3.4 Accès vue	391
4. La sécurité d'une API	392
4.1 Préparer la classe User	397
4.2 Créer la classe Authenticator	398
4.3 Configurer le fichier security.yaml	401
4.4 Ajouter des utilisateurs pour l'accès à l'API	402
4.5 Tester l'accès à l'API	404

Chapitre 18

Personnalisation des pages d'erreurs	407
--	-----

Chapitre 19

L'internationalisation

1. Introduction	411
2. Le principe de la traduction	411
3. La variable locale	412
4. Les catalogues de traduction	414
5. Les éléments à traduire	415
6. Les variables dans les traductions	417
7. L'aide à la mise à jour des catalogues	418
8. L'organisation des catalogues	418
9. La gestion du pluriel	422
10. La traduction des messages des contraintes de validation	425
11. L'utilisation du nom de domaine	427

Chapitre 20

Les services

1. Rappel sur les espaces de noms	429
2. Notion de service	431
3. Utilisation des services	432
4. Création de son propre service	434
5. Injection d'un service dans un service	437

10 _____ Apprendre à développer

des applications web avec PHP et Symfony

Chapitre 21

La classe Swift Mailer

- 1. Installation et configuration 441
- 2. Envoi d'e-mails 444

Chapitre 22

Déployer son site en production

- 1. Introduction 447
- 2. Gestion des performances 448
 - 2.1 Le cache HTTP 448
 - 2.2 Mise en place des caches sur les contrôleurs 449
- 3. Utiliser un environnement de production 453
- 4. Vérifier la sécurité des dépendances 453
- 5. Préparer le serveur de production 454
- 6. Déployer votre application 455
 - 6.1 Déploiement via FTP 456
 - 6.2 Déploiement via des outils open source 460
 - 6.3 Déploiement via le Cloud 460

Conclusion 463

Index 465

Chapitre 5

Les frameworks

1. Un framework : pour quoi faire ?

Nous avons vu précédemment, comment créer du code en PHP et construire une page web.

Il est très possible de développer des applications en utilisant directement le langage PHP.

Alors pourquoi utiliser un framework ?

Un framework est une structure de base préétablie dans laquelle vous allez pouvoir développer.

1.1 Quels sont les avantages ?

- La structure du framework vous permet d'aller plus vite dans le développement du code. Un framework contient la plupart des classes dont vous aurez besoin pour mettre au point votre application. Vous n'avez pas besoin de tout développer vous-même !
- Votre code est déjà structuré. Le découpage en sous-dossiers et fichiers est déjà défini. Vous n'avez plus qu'à vous y conformer. De plus, cette structure est optimale.

- Les développeurs qui travailleront sur votre application utiliseront la même structure, les mêmes standards de code. Les bonnes pratiques sont encouragées. Il est plus facile de travailler à plusieurs lorsqu'on adopte la même logique. Il en est fini des codes de développeurs solitaires, qui étaient difficiles à reprendre et à redévelopper pour les autres.
- Il est plus facile pour un développeur connaissant le framework de plonger dans votre application. Il mettra moins de temps à pouvoir travailler dessus. Vous trouverez facilement sur les sites d'annonces d'emplois un candidat qui maîtrisera le framework.
- Les frameworks les plus célèbres possèdent une large communauté. Vous ne serez plus seul face à votre code.
- Vous pourrez récupérer certains composants de votre code pour d'autres applications. L'utilisation du framework entraîne la réutilisabilité du code.
- Un framework évolue et se maintient dans le temps. Vous bénéficierez de cette évolution. Un petit bémol toutefois, quand les nouvelles versions ne sont plus compatibles avec les anciennes, mais cela arrive rarement.

Comme tout, le framework présente aussi des inconvénients.

1.2 Quels sont les inconvénients ?

- Un framework entraîne le chargement de bibliothèques lourdes et dont vous n'avez pas forcément besoin. Cela peut avoir un impact sur la performance et le temps de réponse. Pour pallier ce problème, les frameworks proposent une version light (allégée) qui vous laisse le soin de charger uniquement les bibliothèques dont vous avez besoin de manière dynamique.
- Vous n'avez plus besoin d'écrire votre propre code. Eh oui, c'est un inconvénient. Beaucoup d'utilisateurs de frameworks aujourd'hui n'ont plus le goût, ni même l'aptitude à développer et c'est dommage. Vous aurez tendance à chercher la solution à votre problème dans les outils du framework ou de la communauté au lieu de les inventer vous-même.

- L'utilisation d'un framework implique une période d'apprentissage (d'où cet ouvrage). On ne plonge pas dans un framework sans en connaître les fondamentaux. C'est donc une étape supplémentaire à maîtriser par rapport au langage PHP.
- Un framework évolue sans arrêt. Il se peut que certaines mises à jour mettent à mal vos applications passées. Il faudra faire évoluer vos applications en fonction des mises à jour.
- Réfléchissez bien au framework que vous allez utiliser et ne vous laissez pas séduire par la popularité d'un framework. On a vu dans le passé des frameworks très populaires qui, cinq ans après, sont devenus obsolètes. La mode change, c'est valable aussi pour les frameworks.

Vous avez, a priori, choisi Symfony et c'est un bon choix au regard de ses possibilités, de son utilisation et de son évolution. Vous n'aurez donc pas ce genre de déconvenues dans l'avenir.

2. Les frameworks PHP

À l'heure où ces lignes sont écrites, il existe trois grands frameworks PHP qui se partagent le marché :

- Symfony, bien sûr, qui est incontestablement le framework le plus apprécié, surtout en Europe. Il possède une très grosse communauté. C'est un framework français (développé par SensioLabs). Son utilisation est grandissante, notamment pour de gros projets comme Drupal 8, eZ Publish 5, Dailymotion, BlablaCar...
- Laravel, le plus gros concurrent de Symfony. Il reprend beaucoup de composants issus de Symfony, comme le système de routage, la gestion des formulaires, les classes de requêtes et de réponses... Il est très utilisé aux États-Unis.
- CodeIgniter, qui est le plus simple des frameworks. On peut l'appréhender en moins d'une heure. C'est ce qui fait sa popularité. Il est utile pour ceux qui veulent avoir une structure de base de framework et rien de plus. Mais ses fonctionnalités sont beaucoup moins développées que celles des autres.

Il existe d'autres frameworks, comme Zend, Yii ou CakePhp qui sont beaucoup moins utilisés aujourd'hui.

3. Le framework Symfony

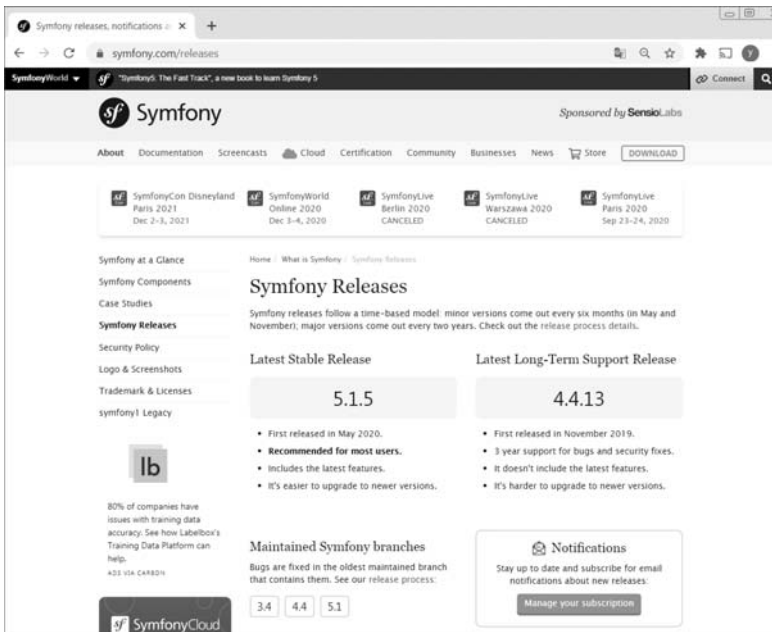
3.1 Présentation du framework

Le framework Symfony est celui qui nous intéresse particulièrement. Sa grande communauté, son adaptabilité, sa longévité en font l'un des frameworks les plus fiables aujourd'hui.

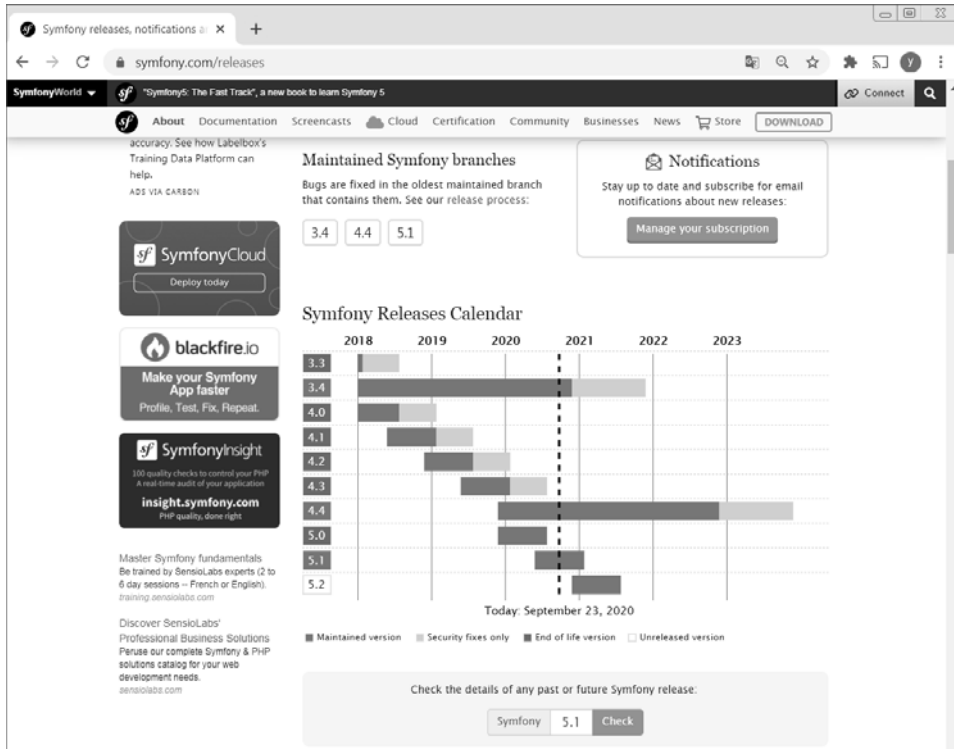
Symfony est en open source, ce qui signifie que vous pouvez le télécharger et l'utiliser gratuitement même pour des applications commerciales. Sa première version est sortie en 2005.

Rendons-nous sur le site de Symfony : <https://symfony.com>

Si vous cliquez sur le menu **About** dans la barre de navigation supérieure, puis sur **Symfony Releases** dans le menu vertical gauche, vous découvrirez la version actuelle de Symfony (*Latest Stable Release*) et la dernière version de support à long terme (*Latest Long-Term Support Release*) :



Si vous scrollez sur la page, vous découvrez la *roadmap*, c'est-à-dire le calendrier des mises à jour de chaque version :



On s'aperçoit que deux versions continuent à être supportées longtemps : la version 3.4 et la version 4.4. Ceci s'explique par le fait qu'un changement important de la structure de base de Symfony a été effectué à partir de la version 4.

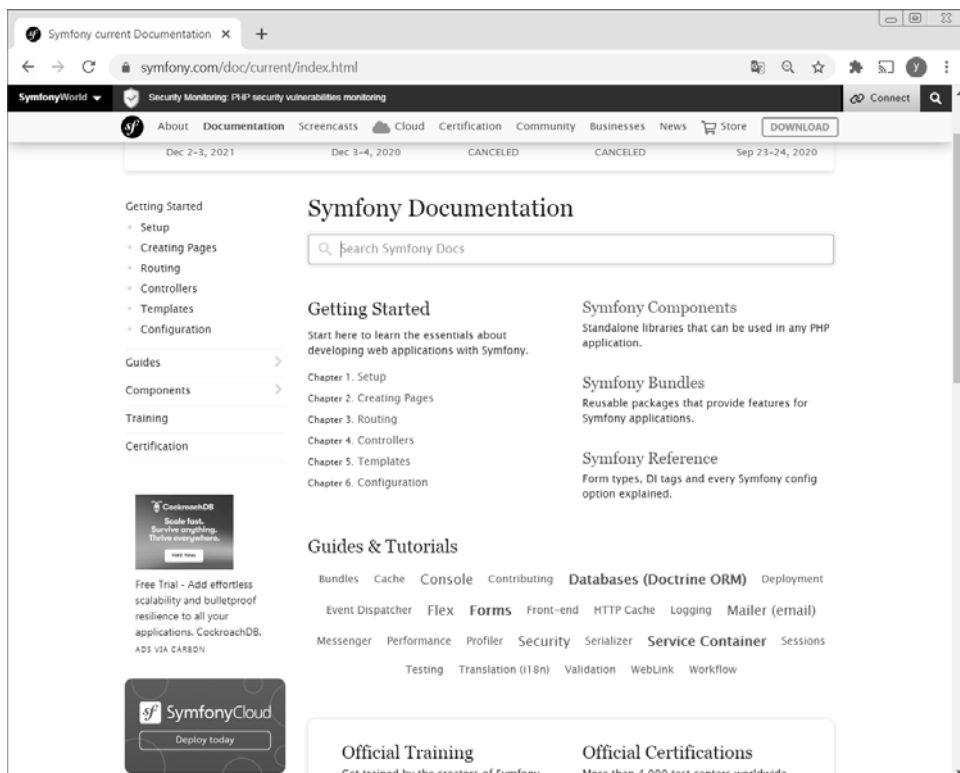
Vos applications en version 3.4 ou antérieure doivent être remaniées pour pouvoir fonctionner dans la version 4. Symfony continue donc à supporter la version 3.4, afin de laisser un peu de temps aux entreprises pour leur permettre de faire leurs migrations (jusqu'en 2022).

Nous allons utiliser la version actuelle, la version 5, qui ne présente pas de modifications majeures par rapport à la version 4.

3.2 La documentation

La documentation est très bien faite en Symfony. Nous vous engageons vivement à vous y référer le plus souvent possible. Le premier réflexe lorsqu'on a besoin d'une solution en Symfony (avant de faire une recherche sur Internet) est de consulter la documentation.

■ Cliquez sur l'onglet **Documentation**. Vous avez une zone de recherche qui vous permet de rechercher à l'aide de mots-clés des informations dans la documentation.



Vous disposez de plusieurs sortes de documentation :

Le Quick Tour : vous le trouvez en cliquant sur l'onglet **About** et en scrollant vers le bas. C'est un tour rapide d'utilisation de Symfony. Vous y apprendrez comment installer Symfony et les fondamentaux pour son utilisation basique. Il vous permettra de vous faire une opinion du framework sans trop vous impliquer. Mais cela reste une découverte très rapide du framework.

Le Getting Started : vous le trouvez en cliquant sur l'onglet **Documentation**. Avec ses six chapitres, il vous permet d'approfondir votre connaissance du framework pour découvrir l'essentiel du développement d'applications web.

Le Symfony Components : vous le trouvez en cliquant sur l'onglet **Documentation**. Symfony est un framework composé de différents composants indépendants (les composants). Il est possible d'installer un seul composant, ponctuellement, dans un script PHP, sans installer la totalité du framework. Cette documentation vous montre comment installer chaque composant de manière isolée et comment l'utiliser.

Le Symfony Reference : vous le trouvez en cliquant sur l'onglet **Documentation**. C'est une documentation beaucoup plus technique et poussée. Elle vous donnera des informations détaillées sur la configuration du framework et des détails sur certains points, comme les formulaires, les services... Elle s'adresse à des développeurs qui maîtrisent déjà le framework.

Le Symfony Framework Best Practices : vous le trouvez en cliquant sur l'onglet **Documentation**. Comme son nom l'indique, ce sont les bonnes pratiques d'utilisation du framework. C'est intéressant de connaître les bases du bon développement du framework telles que les ont imaginées les créateurs (en l'occurrence, Fabrice Potentier, créateur de Symfony).

Le Symfony CMF : vous le trouvez en cliquant sur l'onglet **Documentation** en bas de la page.

Symfony CMF est un CMS conçu par la communauté à partir de Symfony. Un CMS est un gestionnaire de contenu utilisable en mode graphique sans avoir à développer du code (comme WordPress, par exemple). C'est intéressant par curiosité, mais inutile si vous voulez développer votre propre application Symfony.