

Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence ENI de l'ouvrage **EI3SYM** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Préface

Chapitre 1

Avant-propos

1. Introduction	21
2. Public visé	22
3. Pourquoi un framework ?	22
3.1 header() et echo	23
3.2 Éviter la globalité	23
3.3 Ne pas réinventer la roue	24
4. Pourquoi Symfony ?	24
5. Prérequis	25
6. Objectifs du livre	25

Chapitre 2

Architecture du framework

1. Le patron de conception MVC	27
1.1 Définitions	27
1.1.1 La vue	27
1.1.2 Le modèle	28
1.1.3 Le contrôleur	28

1.2	En pratique	28
1.2.1	Le contrôleur frontal	29
1.2.2	Le routage	29
1.2.3	Le contrôleur et le modèle	30
1.2.4	La vue	31
2.	Architecture de Symfony	31
2.1	Schéma	32
2.2	Le Service Container	33
2.3	Un framework MVC ?	34
2.4	Une flexibilité à toute épreuve	34
3.	Les bundles	35
3.1	Concept	35
3.2	Un écosystème mature	36
4.	Les environnements	37
4.1	Principe	37
4.2	En pratique	37
4.2.1	Contexte HTTP	37
4.2.2	Contexte CLI (Command Line Interface)	37
4.2.3	Exemples de différences selon l'environnement	38

Chapitre 3

Débuter avec Symfony

1.	Créer un projet sous Symfony	39
1.1	L'édition « standard »	39
1.2	Prérequis	40
1.3	Installation via l'installeur Symfony (Linux et Mac OS)	40
1.4	Installation via Composer	41
1.4.1	Installer Composer	41
1.4.2	Créer un projet	45
1.4.3	Les versions	46

- 2. Découvrir Symfony 48
 - 2.1 Configurer son serveur web 48
 - 2.1.1 Serveur web PHP 48
 - 2.1.2 Apache et Nginx 49
 - 2.2 Structure de l'application 53
- 3. La console 54
 - 3.1 Emplacement 54
 - 3.2 Les commandes 55
 - 3.2.1 Lister les commandes disponibles 55
 - 3.2.2 Exécuter une commande 56
 - 3.3 Les options 56
 - 3.4 Les arguments 57
 - 3.5 La commande help 57
 - 3.6 Exécuter rapidement des commandes 59
 - 3.6.1 Les raccourcis 59
 - 3.6.2 L'autocomplétion 59
- 4. L'autochargement des classes 60
 - 4.1 Le standard PSR-4 61
 - 4.2 Autres mécanismes 62
 - 4.3 Le fichier vendor/autoload.php 62
- 5. Installer un bundle 63
 - 5.1 Bundle applicatif spécifique au projet 63
 - 5.1.1 Présentation 63
 - 5.1.2 Créer un bundle 64
 - 5.2 Bundle tiers 65

Chapitre 4 **Routage et contrôleur**

1. Comprendre le routage	69
1.1 Définition	69
1.2 Le répertoire web et le contrôleur frontal	70
1.3 Une requête, une action	70
2. Format des routes	71
2.1 Différentes méthodes	71
2.2 Les annotations	72
3. Configurer le path	73
3.1 Le path /hello/world	73
3.2 Comprendre la notation du contrôleur	77
3.3 Choisir son format	78
3.4 Importer ses routes depuis des fichiers externes	79
3.5 Utiliser des paramètres de substitution	80
3.6 Comprendre l'ordre de chargement des routes	83
3.7 Préfixer les routes	83
3.8 Ajouter des restrictions	85
4. Routage par nom de domaine	87
4.1 Prérequis	88
4.2 Exemple	88
4.3 Explications	91
5. Le contrôleur	92
5.1 Récupérer un service	93
5.2 Utiliser les « paramètres magiques »	93
5.2.1 Paramètres de substitution des routes	93
5.2.2 Exemples	94
5.3 Générer une URL	95
5.4 Effectuer une redirection	96
5.5 Redirection interne	97

5.6 Renvoyer des pages d'erreurs	97
5.6.1 Le contrôleur	98
5.6.2 La vue	100

Chapitre 5

L'injection de dépendances

1. Une alternative au Singleton	105
1.1 Présentation du Singleton	105
1.2 Exemple	106
1.3 Un patron de conception à utiliser avec précaution	107
2. L'injection de dépendances	108
2.1 L'injection de dépendances par le constructeur	108
2.2 L'injection de dépendances par setter (mutateur)	109
2.3 L'injection de dépendances par propriété	110
2.4 Les avantages	111
2.5 Une exécution complexifiée	111
3. Le Service Container	112
3.1 Les services	112
3.2 Explications au travers d'un service X	113
4. Créer un service et configurer ses injections	114
4.1 Créer un service	114
4.2 Injection par constructeur	114
4.3 Injection par méthode	115
4.4 Injection par propriété	116
4.5 Injection automatique avec l'autowiring	116
4.6 Utiliser des paramètres	118
4.7 Créer des services « lazy »	120
5. Les extensions de bundle	121
5.1 Le dossier DependencyInjection	123
5.2 Les définitions de services depuis un bundle	123

6 **Symfony 3**

Développez des sites web PHP structurés et performants

5.3	La configuration	124
5.3.1	Définir une arborescence	125
5.3.2	Les différentes étapes du traitement de la configuration	131
5.3.3	Récupérer la configuration validée	134
5.4	Les « Compiler Passes »	138
5.4.1	Concept	138
5.4.2	Les tags	141
5.4.3	Le Compiler Pass.	142

Chapitre 6 **Les templates avec Twig**

1.	Un langage accessible	145
1.1	Aperçu	145
1.2	Pourquoi un nouveau langage ?	146
1.3	Mise en pratique	147
1.4	Remarques sur l'utilisation	148
1.5	Notations des templates	149
1.6	L'annotation @Template.	150
2.	Layouts (gabarits de pages) et blocks	152
3.	Structures de contrôle et tags	160
3.1	Les conditions	160
3.2	Les boucles	161
3.3	Créer et modifier des variables.	163
3.4	Twig et l'échappement	164
3.4.1	Le tag autoescape	165
3.4.2	Échappement unitaire	166
3.4.3	Modifier la stratégie globale (utilisateurs avancés)	167
3.5	Inclure des templates	167

- 4. Les filtres et les fonctions. 168
 - 4.1 Les filtres. 168
 - 4.1.1 Utilisation et syntaxe. 168
 - 4.1.2 Chaînes de caractères. 169
 - 4.1.3 Échappement 170
 - 4.1.4 L’encodage. 171
 - 4.2 Les fonctions. 172
 - 4.2.1 Twig et le routage. 173
 - 4.2.2 Débogage avec la fonction dump. 174
- 5. Ressources publiques (images, feuilles de style, scripts JS...) 174
 - 5.1 Comment ces ressources sont-elles synchronisées ? 175
 - 5.2 Référencer les ressources publiques depuis un template 176

Chapitre 7
Les bases de données avec Doctrine2

- 1. Concepts de DBAL, entité et ORM. 177
 - 1.1 DBAL. 178
 - 1.2 Entité. 179
 - 1.3 ORM 179
- 2. Installer et configurer DoctrineBundle 180
 - 2.1 Installation 180
 - 2.2 Configuration 181
- 3. Les entités et leur mapping 182
 - 3.1 Cas pratique : répertoire des livres. 182
 - 3.1.1 L’entité 182
 - 3.1.2 Le mapping 184
 - 3.2 Définir une entité avec @ORM\Entity 187
 - 3.3 Gérer les colonnes de la table avec @ORM\Column. 189
 - 3.3.1 name 189
 - 3.3.2 type 190
 - 3.3.3 length. 191

3.3.4	unique	191
3.3.5	nullable	192
3.3.6	precision et scale	192
3.4	@ORM\Table	192
3.5	Les clés primaires	194
3.6	Configurer les index	195
3.6.1	@ORM/Index	195
3.6.2	@ORM\UniqueConstraint	196
3.7	Les relations (clés étrangères)	196
3.7.1	@ORM\OneToOne	197
3.7.2	@ORM\ManyToOne	199
3.7.3	@ORM\ManyToMany	200
3.7.4	Relations bidirectionnelles	202
3.7.5	Repérer les erreurs de mapping	206
4.	Utiliser l'EntityManager	208
4.1	Insertion de données	208
4.2	Modification de données	210
4.3	Suppression de données	211
4.4	Autres actions de l'EntityManager	212
4.4.1	refresh()	212
4.4.2	detach()	213
4.5	Les opérations en cascade	214
5.	Les repositories et le DQL	218
5.1	Les repositories	218
5.1.1	Un rôle de centralisateur	218
5.1.2	Les méthodes de base du repository	219
5.1.3	Les méthodes personnalisées du repository	221
5.2	Le DQL	222
5.2.1	SELECT	224
5.2.2	FROM	224
5.2.3	JOIN et LEFT JOIN	225
5.2.4	WHERE	228
5.2.5	ORDER BY	233

5.2.6 Les limites	234
5.2.7 Les limites et la pagination	235
5.3 Le QueryBuilder	239
6. Quelques astuces	242
6.1 Utilisation du @ParamConverter	242
6.2 Les extensions Doctrine	244
6.2.1 Installation	244
6.2.2 Utilisation d'un slug sur une entité	245

Chapitre 8

Le répartiteur d'événements

1. Une histoire d'écoute	249
1.1 Le dispatching	250
1.2 Les listeners	253
2. Les événements du Kernel	257
2.1 Les événements	257
2.2 Exemple : effectuer des tâches asynchrones	258
3. Les événements de la console	260
3.1 Prérequis	260
3.2 Les événements	260

Chapitre 9

Utiliser les formulaires

1. Une librairie MVC	263
1.1 Le modèle	263
1.2 Le contrôleur	265
1.3 La vue	266
2. Fonctionnement du composant	267
2.1 L'objet « Form »	267
2.1.1 Soumission	267

2.1.2	Validation	268
2.1.3	Vue	268
2.2	Les types	268
2.3	Les options	269
2.4	Les objets « Form » et « FormBuilder »	270
2.4.1	Le FormBuilder	270
2.4.2	Structure de l'objet Form	270
2.5	Le mapping avec l'objet de la couche Modèle	272
2.6	Les différentes représentations des valeurs	274
2.6.1	Transformation des données	274
2.6.2	Illustration avec le type Date	275
3.	Les types	276
3.1	L'héritage	276
3.2	FormType	277
3.2.1	label	277
3.2.2	label attr	277
3.2.3	data	277
3.2.4	required	277
3.2.5	disabled	278
3.2.6	mapped	278
3.2.7	property_path	278
3.2.8	attr	279
3.2.9	trim	279
3.2.10	error_bubbling	279
3.3	TextType	280
3.4	PasswordType	280
3.5	RepeatedType	281
3.5.1	type	281
3.5.2	first_options et second_options	281
3.5.3	options	281
3.5.4	first_name	282
3.5.5	second_name	282
3.5.6	invalid_message	282

3.6	ChoiceType	282
3.6.1	choices	282
3.6.2	expanded et multiple	283
3.6.3	placeholder	283
3.6.4	preferred_choices	284
3.6.5	Types similaires	284
3.7	EntityType	284
3.7.1	class	285
3.7.2	choice_label	285
3.7.3	query_builder	285
3.7.4	group_by	286
3.7.5	em	286
3.8	DateType	286
3.8.1	widget	286
3.8.2	format	287
3.8.3	model_timezone	287
3.8.4	view_timezone	287
3.8.5	years	288
3.8.6	months	288
3.8.7	days	288
3.8.8	placeholder	288
3.8.9	Types similaires	288
3.9	FileType	289
3.9.1	multiple	289
3.9.2	Récupérer les fichiers	289
3.9.3	Traiter les fichiers	290
3.10	CheckboxType	291
3.11	SubmitType, ResetType et ButtonType	292
4.	Validation des données	293
4.1	Objectif	293
4.2	Utilisation	294
4.2.1	Contraintes	294
4.2.2	Configuration des contraintes	294

4.2.3	Les différents formats de configuration	301
4.2.4	Les options	304
4.2.5	Validation d'un objet hors du contexte d'un formulaire	307
4.3	Liste des contraintes et de leurs options	307
4.3.1	NotBlank et NotNull	308
4.3.2	IsNull et Blank	308
4.3.3	IsTrue, IsFalse	309
4.3.4	Type	309
4.3.5	Email, Url et Ip	310
4.3.6	Regex	311
4.3.7	Length, Count	312
4.3.8	Range	313
4.3.9	Comparaisons	314
4.3.10	Dates	315
4.3.11	File	315
4.3.12	Image	316
4.3.13	Choice	316
4.3.14	UniqueEntity	317
4.3.15	Données financières	318
4.3.16	Callback	319
4.3.17	All	320
4.3.18	Valid	320
4.4	Groupes de validation	321
5	Créer des formulaires réutilisables	323
5.1	La classe AbstractType	323
5.2	Utiliser un formulaire défini dans une classe	330
5.2.1	Instanciation manuelle	330
5.2.2	Avec l'injection de dépendances	330
6	Personnaliser le rendu - thèmes de formulaires	332
6.1	Afficher le formulaire manuellement	332
6.1.1	form_start()	333
6.1.2	form_end()	333

- 6.1.3 form_widget() 334
- 6.1.4 form_errors() 334
- 6.1.5 form_label() 335
- 6.1.6 form_row() 335
- 6.1.7 form_rest() 335
- 6.1.8 form_enctype() 335
- 6.1.9 Arborecence des parties de formulaires 335
- 6.2 Créer des thèmes 337
 - 6.2.1 Formulaire d'exemple 337
 - 6.2.2 Créer et associer un thème de formulaires 338
 - 6.2.3 Comprendre le nom des blocks 342

Chapitre 10
La sécurité

- 1. Présentation 345
- 2. Authentification 347
 - 2.1 Pare-feu 347
 - 2.2 Pare-feu pour ressources statiques/développement 348
 - 2.3 Authentification HTTP 349
 - 2.4 Authentification par formulaire de connexion 349
 - 2.5 Déconnexion 353
- 3. Utilisateurs et rôles 354
 - 3.1 Récupérer l'utilisateur courant 354
 - 3.2 L'utilisateur 355
 - 3.3 Les fournisseurs d'utilisateurs 356
 - 3.3.1 En mémoire 356
 - 3.3.2 Fournisseur d'utilisateurs de bases de données 357
 - 3.3.3 Fournisseur d'utilisateurs personnalisé 359
 - 3.3.4 Notes additionnelles 363
 - 3.4 Cryptage des mots de passe 364
 - 3.4.1 Encodeurs 364
 - 3.4.2 Le salage 365

3.4.3	Crypter un mot de passe	367
3.5	Les rôles	368
4.	Autorisations	371
4.1	Les rôles, au cœur du processus	371
4.2	Vérifier le rôle de l'utilisateur	372
4.3	Sécuriser une action	373
4.4	Sécuriser une section de l'application	374
4.5	Sécuriser selon d'autres critères	375
4.6	Pour aller plus loin	377

Chapitre 11

Logging et monitoring

1.	Créer des logs avec Monolog	379
1.1	Journalisation	379
1.2	Monolog	380
1.2.1	Le standard PSR-3	380
1.2.2	MonologBundle	381
1.3	Le service logger	381
1.4	Le fichier de logs	382
1.4.1	Identifier la cause d'un bogue	382
1.4.2	Le problème	382
1.5	Les gestionnaires (handlers)	383
1.5.1	Définir plusieurs gestionnaires	383
1.5.2	Envoyer des logs par e-mail	384
1.5.3	Utiliser un tampon (buffer)	385
1.5.4	Ajouter des informations complémentaires	385
1.6	Les canaux (channels)	387
1.6.1	Ajouter ses propres canaux	387
1.6.2	Envoyer un enregistrement sur un canal donné	388
1.6.3	Configurer les gestionnaires par canaux	388
1.6.4	Gestion des erreurs 404	389

- 2. Le monitoring avec Prometheus et Grafana 391
 - 2.1 Un allié proactif au logging 391
 - 2.2 Préparation d'une application Symfony pour Prometheus ... 392
 - 2.3 Instrumentaliser les mesures 397
 - 2.4 Pour aller plus loin 398

Chapitre 12

Tester son application Symfony

- 1. Les tests unitaires et fonctionnels 399
 - 1.1 Concept 399
 - 1.1.1 L'automobile 400
 - 1.1.2 Les tests sous Symfony 400
 - 1.2 Installation de PHPUnit 401
- 2. Tests unitaires 401
 - 2.1 Exécuter les tests 403
 - 2.2 Exécuter une partie des tests 403
- 3. Tests fonctionnels 404
 - 3.1 Tester une action 404
 - 3.2 L'objet Client 405
 - 3.3 L'objet Crawler 408
 - 3.4 Soumettre un formulaire 410
 - 3.5 Pour aller plus loin 411

Chapitre 13

Améliorer les performances de son application

- 1. La mise en cache de pages 413
 - 1.1 Autour du protocole HTTP 413
 - 1.2 Le proxy inverse (ou « reverse proxy ») 414
 - 1.2.1 HttpCache 416
 - 1.2.2 Nginx 417
 - 1.2.3 Varnish 418

1.3	Les en-têtes	423
1.4	Les réponses publiques et privées	423
1.5	L'expiration	424
1.5.1	L'en-tête Expires	424
1.5.2	Les directives max-age et s-max-age	425
1.5.3	L'annotation @Cache	425
1.6	La validation	427
1.6.1	Par date avec Last-Modified	428
1.6.2	Par empreinte avec l'en-tête ETag	429
1.7	Les ESI	430
1.7.1	Activation	431
1.7.2	Générer une balise ESI	431
2.	L'autochargement des classes	433
2.1	Générer un classmap	433
2.2	Englober le chargeur de classe de Composer	433
3.	Le cache avec Doctrine	434
3.1	Les différents types de cache	434
3.1.1	Le cache des métadonnées	434
3.1.2	Le cache des requêtes	434
3.1.3	Le cache des résultats	434
3.2	Configuration	435
4.	Le cache d'annotations	436
5.	Les sessions	437
6.	L'extension PHP pour Twig	438
6.1	Installation via PEAR	438
6.2	Installation depuis les sources	438
7.	Autres optimisations	439
7.1	Choix de sa SAPI PHP	439
7.1.1	Qu'est-ce qu'une SAPI ?	439
7.1.2	Module du serveur	440
7.1.3	CGI	440
7.1.4	FastCGI	441

- 7.1.5 Conclusion 441
- 7.2 Mise en cache d'OPCodes. 441
 - 7.2.1 Les OPCodes 442
 - 7.2.2 Une étape lourde. 442
 - 7.2.3 La mise en cache 443
- 7.3 La compression des réponses 443
 - 7.3.1 Compression gzip 443
 - 7.3.2 Précompression 444
- 7.4 Optimisation des images 444
 - 7.4.1 Validation 445
 - 7.4.2 Expiration 445
 - 7.4.3 Autres techniques 445
- 7.5 Conseils de Google 446
- 8. Test des performances d'un site web. 446
 - 8.1 Côté serveur 446
 - 8.1.1 Apache Bench 446
 - 8.1.2 Xhprof 447
 - 8.2 Côté client. 447

Annexes

- 1. Développer son projet Symfony sur une machine virtuelle. 449
 - 1.1 Introduction à la virtualisation 449
 - 1.2 Virtualisation et développement web 450
 - 1.3 Installation 451
 - 1.3.1 Configuration 452
 - 1.3.2 Utilisation 453
- 2. Créer une commande pour la console 454
 - 2.1 La configuration d'une commande 454
 - 2.2 Les objets input et output 456
 - 2.3 Le Service Container 458
 - 2.4 Commande d'exemple 459

3.	Envoyer des e-mails grâce à SwiftMailer	460
3.1	Le protocole SMTP	460
3.2	Le transport	460
3.2.1	Le transport smtp	461
3.2.2	Le transport sendmail	464
3.2.3	Le transport mail	465
3.2.4	Choisir son transport	466
3.3	Envoi d'un e-mail	467
3.4	Le spool d'e-mails	468
3.5	Pendant le développement	470
4.	Gérer ses utilisateurs avec FOSUserBundle	471
4.1	Installation	471
4.2	Aperçu des fonctionnalités	472
4.3	Activation des traductions	473
4.4	Configuration	474
4.4.1	Création de la classe utilisateur	474
4.4.2	Configuration de la sécurité	475
4.4.3	Configuration du bundle	475
4.5	Importation des routes	476
4.6	Inscription des utilisateurs	477
4.6.1	Accéder au formulaire d'inscription	477
4.6.2	Personnaliser le formulaire avec un CAPTCHA	477
4.6.3	Envoi d'un e-mail de confirmation	480
4.7	Pour aller plus loin	480
5.	Les traductions	481
5.1	Introduction	481
5.1.1	La culture (Locale)	481
5.1.2	Internationalisation	482
5.1.3	Régionalisation	482
5.2	Détecter la culture d'un utilisateur	482
5.2.1	Différentes possibilités	482
5.2.2	En pratique	483
5.3	Deux principaux cas d'utilisation des traductions	484

5.4	Activation des traductions	484
5.5	Routes multilingues	485
5.6	JMSI18nRoutingBundle	485
5.6.1	Installation	486
5.6.2	Configuration de la stratégie	486
5.6.3	Traductions des routes	487
5.7	Les fichiers de traductions	488
5.7.1	Emplacement	488
5.7.2	Règle de nommage	488
5.8	Traduction d'un message	490
5.8.1	Le service translator	490
5.8.2	Les paramètres de substitution (placeholders)	491
5.8.3	Dans les templates Twig	491
6.	Travailler avec les sessions	492
6.1	Introduction	492
6.2	Intégration des sessions dans Symfony	493
6.3	Configuration du gestionnaire de sauvegarde	493
6.3.1	Avec PHP	493
6.3.2	Avec Symfony	494
6.4	Les messages « flash »	495
7.	Déployer son application	496
7.1	Le déploiement	496
7.2	Faut-il déployer par FTP ?	497
7.3	Les différentes étapes	498
7.4	Capistrano et Capifony	499
7.4.1	Installation	499
7.4.2	Configuration	499
7.4.3	Déploiement	501
7.5	Fonctionnalités avancées	502
	Index	503