

Cheat Sheet — Commandes Symfony

Commandes Symfony avec Claude-Craft

The Bearded Bear

Février 2026



Cheat Sheet : Commandes Symfony Claude-Craft

Docker : Toutes les commandes utilisent `docker compose exec app` pour s'abstraire de l'environnement local.

Commandes Symfony (10)

Generation de code (3)

```
# CRUD complet (Create, Read, Update, Delete)
/symfony:generate-crud Category

# Handler CQRS
/symfony:generate-command CreateOrder \
  customerId:uuid items:array

# Endpoint API Platform
/symfony:api-endpoint
```

Audit et verification (5)

Commande	Description
/symfony:check-compliance	Audit DDD complet
/symfony:check-architecture	Validation Clean Architecture
/symfony:check-code-quality	PHPStan, CS-Fixer, complexite
/symfony:check-security	OWASP Top 10, injections
/symfony:check-testing	Couverture, qualite tests

Outils qualite (via Docker)

```
# PHPStan
docker compose exec app \
  php vendor/bin/phpstan analyse --level=8

# PHP-CS-Fixer (verification)
docker compose exec app \
  php vendor/bin/php-cs-fixer fix --dry-run --diff

# PHP-CS-Fixer (correction)
docker compose exec app \
  php vendor/bin/php-cs-fixer fix
```

Base de donnees (2)

Commande	Description
/symfony:migration-plan	Planifier migration DB
/symfony:optimize-doctrine	Optimiser requetes Doctrine

Tests (via Docker)

```
# Tests unitaires
docker compose exec app \
  ./vendor/bin/phpunit tests/Unit/

# Tests integration
docker compose exec app \
  ./vendor/bin/phpunit tests/Integration/

# Avec couverture
docker compose exec app \
  ./vendor/bin/phpunit --coverage-text
```

Agents utiles

Agent	Usage
@symfony-reviewer	Review code Symfony
@tdd-coach	Guide TDD
@api-designer	Conception REST/GraphQL
@database-architect	Schema, optimisation
@refactoring-specialist	Refactoring guide
@performance-auditor	Analyse performance

Commandes Common utiles

Commande	Description
/common:pre-commit-check	Valider avant commit
/common:architecture-decision	ADR
/common:setup-project-context	Config contexte projet
/common:ralph-run	Boucle IA continue
/workflow:init	Initialiser workflow

Raccourcis frequents

Besoin	Action
Nouvelle feature	/workflow:init+ /symfony:generate-crud
Audit rapide	/symfony:check-compliance
Tests	@tdd-coach+ /symfony:check-testing
Review	@symfony-reviewer
Securite	/symfony:check-security
Refactoring	@refactoring-specialist
ADR	/common:architecture-decision

Exemple : Nouvelle feature

```
# 1. Initialiser workflow
/workflow:init "Systeme de panier"

# 2. Design
@api-designer "API panier"
@database-architect "Schema Cart, CartItem"

# 3. Generation
/symfony:generate-crud Cart
/symfony:generate-command AddToCart \
  cartId:uuid productId:uuid quantity:integer

# 4. Tests (via Docker)
@tdd-coach "Tests pour AddToCartHandler"
docker compose exec app \
```

```
./vendor/bin/phpunit tests/Unit/

# 5. Verification
/symfony:check-compliance
docker compose exec app \
  php vendor/bin/phpstan analyse --level=8
```

Exemple : Audit projet existant

```
# 1. Audit complet
/symfony:check-compliance

# 2. Focus securite
/symfony:check-security

# 3. Plan de remediation
"Genere un plan pour les issues critiques"

# 4. Re-verification
/symfony:check-compliance

# 5. Tests finaux (via Docker)
docker compose exec app ./vendor/bin/phpunit
docker compose exec app \
  php vendor/bin/phpstan analyse --level=8
```

Tips

- Commencer par `/workflow: init` pour les features
 - Utiliser les agents pour les decisions d'architecture
 - Toujours verifier avec `/symfony:check-compliance`
 - Les commandes `generate-*` incluent les tests
 - **Toujours utiliser `docker compose exec app`**
-

Formation Claude Code 2.1.45 + Claude-Craft 7.13.0 The Bearded CTO - 2026