

Docs » Symfony Bundles » Bundle Skeleton

Bundle Skeleton

Reusable Symfony bundle template with semantic configuration, explicit service wiring and real kernel tests

Anonymous · 26/10/2026 · #php, #symfony

Symfony Bundle Skeleton

`derafu/symfony-bundle-skeleton` is the base for reusable Derafu bundles targeting PHP 8.5 and Symfony 8.1. Its Composer type is `symfony-bundle`.

Create a new bundle with:

```
composer create-project derafu/symfony-bundle-skeleton my-bundle --stability=dev
```

Structure

```
src/  
  ProjectBundle.php  
  Service/GreetingService.php  
resources/config/  
  symfony-project-bundle-services.yaml  
tests/  
  fixtures/Kernel.php  
  src/ProjectBundleTest.php  
  src/ServicesTest.php
```

`ProjectBundle` extends Symfony's `AbstractBundle`. Its `configure()` hook defines the configuration tree, and `loadExtension()` imports the YAML and exposes the validated options as container parameters. The bundle root is the package directory, including when the bundle class lives in `src/`.

The example configuration is:

```
# config/packages/derafu_project.yaml  
derafu_project:  
  message: 'Hello from the application.'
```

The default message is `Hello from the project bundle`. Empty messages, non-string values and unknown options are rejected by Symfony's configuration system.

The service `derafu_project.greeting` receives the message. Its alias

`Derafu\ProjectBundle\Service\GreetingService` allows consumer services to request the class through dependency injection. Both service and alias are private by default.

`GreetingService::greet()` returns the configured message.

Register the bundle in the consuming application's `config/bundles.php`:

```
return [  
    Derafu\ProjectBundle\ProjectBundle::class => ['all' => true],  
];
```

Adapt the Template

Rename the Composer package and metadata, `Derafu\ProjectBundle` and `Derafu\TestsProjectBundle` namespaces, the bundle class, the `derafu_project` configuration alias, parameter and service prefixes, and the YAML filename together. Update the import in `loadExtension()` and the test kernel's bundle registration and configuration accordingly.

For an actual bundle named `derafu/symfony-example-bundle`, use a filename such as `resources/config/symfony-example-bundle-services.yaml`. Keep the same base structure, configuration hooks, explicit wiring and verification conventions across bundles.

Replace the greeting example with the bundle's actual integration. Add compiler passes, assets, translations or other resources when the feature requires them. Form, Twig and Derafu Translation are not dependencies of this template.

Verification

`ServicesTest` imports the real YAML, supplies the consumer's message parameter, compiles the container and verifies the configured service and alias. Tested definitions are made reachable during compilation so private-service removal cannot conceal invalid wiring. This YAML declares no tags; add tag-consumption tests when the actual bundle introduces them.

`ProjectBundleTest` boots a real application with `FrameworkBundle` and the project bundle. It verifies registration, root path, configuration alias, default and custom messages, and rejection of invalid options. The fixture keeps generated cache and logs in an isolated temporary project and restores environment changes made by Symfony's debug kernel.

Run from the package root:

```
composer update  
composer validate --strict  
composer tests  
composer phpstan  
composer phpcs
```

PHPStan runs at level 9 over `src/` and `tests/`. PHPUnit requires coverage metadata and fails on warnings, notices, deprecations and risky tests. Coverage and test reports are written under `var/`. The CI workflow runs the same checks on PHP 8.5.

Last updated on 26/10/2026