

Progression sur la quête **Générer le CRUD**

(/tasks/7172)

(/tasks/7173)

(/tasks/7174)

66%

## Générer le CRUD | Génération du CRUD

Il ne nous reste plus qu'à générer le CRUD, alors allons-y. Entre la commande suivante dans le terminal :

```
php app/console doctrine:generate:crud
```

Voilà le résultat de la commande :

```
romain@romain-ThinkPad-T420s: ~
romain@romain-ThinkPad-T420s:/var/www/html/flyaround$ php app/console doctrine:generate:crud
PHP Warning:  Module 'xdebug' already loaded in Unknown on line 0

Welcome to the Doctrine2 CRUD generator

This command helps you generate CRUD controllers and templates.

First, give the name of the existing entity for which you want to generate a CRUD
(use the shortcut notation like AcmeBlogBundle:Post)

The Entity shortcut name: WCSCoavBundle:PlaneModel

By default, the generator creates two actions: list and show.
You can also ask it to generate "write" actions: new, update, and delete.

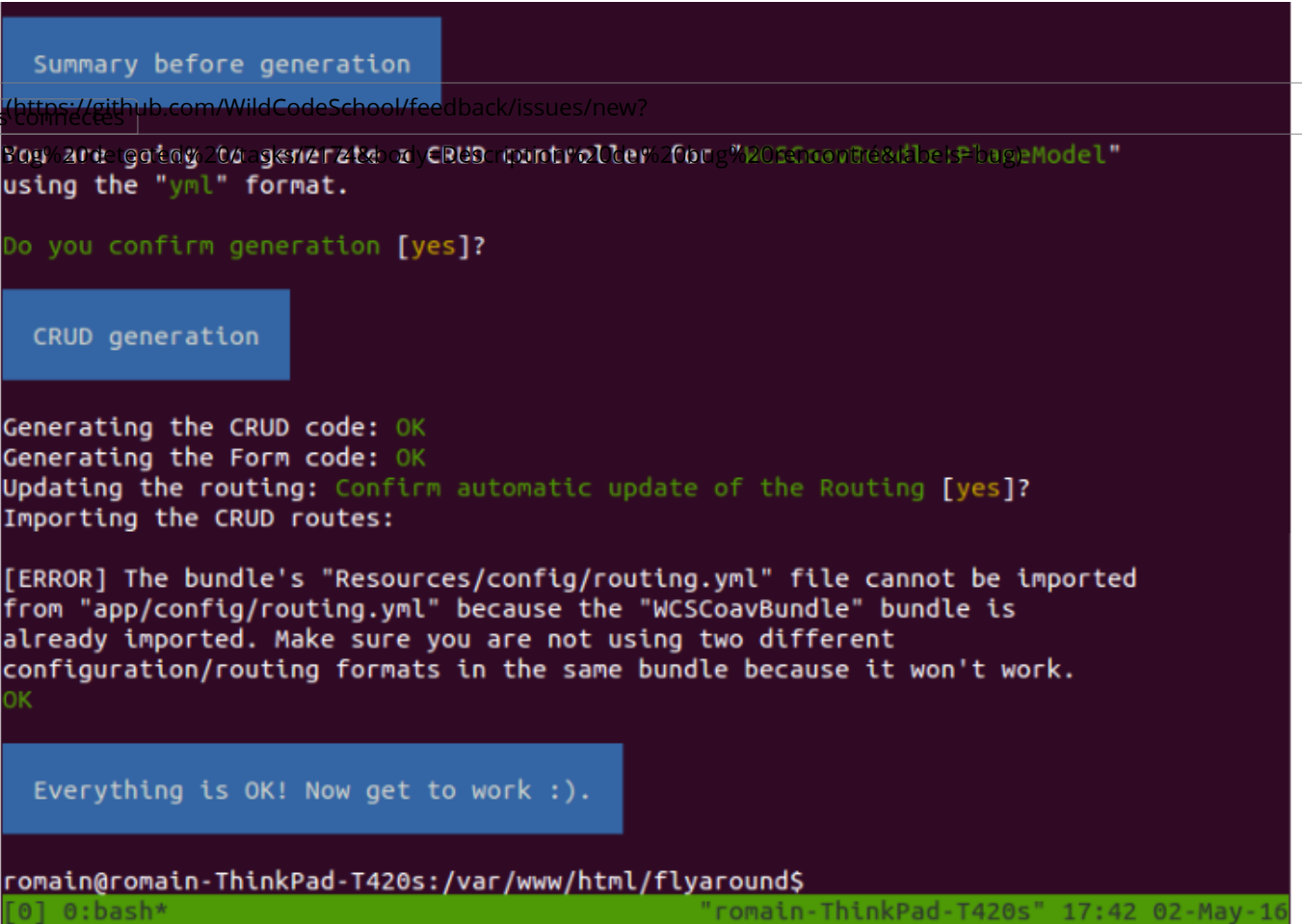
Do you want to generate the "write" actions [no]? yes

Determine the format to use for the generated CRUD.

Configuration format (yaml, xml, php, or annotation) [annotation]: yaml

Determine the routes prefix (all the routes will be "mounted" under this
prefix: /prefix/, /prefix/new, ...).

Routes prefix [/planemodel]:
```



```
Summary before generation

(https://github.com/WildCodeSchool/feedback/issues/new?
éléments connectés
title=Bug%20regard%20tasks/7174&body=Description%20du%20bug%20rencontré&labels=bug)

Generating the CRUD code: OK
Generating the Form code: OK
Updating the routing: Confirm automatic update of the Routing [yes]?
Importing the CRUD routes:

[ERROR] The bundle's "Resources/config/routing.yml" file cannot be imported
from "app/config/routing.yml" because the "WCSCoavBundle" bundle is
already imported. Make sure you are not using two different
configuration/routing formats in the same bundle because it won't work.
OK

Everything is OK! Now get to work :).

romain@romain-ThinkPad-T420s:/var/www/html/flyaround$
[0] 0: bash* "romain-ThinkPad-T420s" 17:42 02-May-16
```

Commentons ce résultat :

- Le CRUD s'applique forcément à une entité, donc la première étape consiste à préciser à Symfony l'entité sur laquelle il génèrera le CRUD. Attention ! Il faut que cette entité ait été créée en amont.
- Il nous demande ensuite si l'on veut générer les actions d'écriture. Il parle des action du controleur qu'il va générer. Si l'utilisateur doit pouvoir créer des entités, il faut lui générer ces actions d'écriture. Dans notre cas les utilisateurs pourront créer des model d'avion donc on aura besoin des actions d'écriture.
- Pour la config c'est toujours la même chose. Il nous parle ici de la config des routes.
- Le préfix des routes sert à la construction des URL : /planemodel/, /planemodel/{id}/show, /planemodel/{id}/new

C'est quoi cette erreur qu'il affiche avant de dire OK ?

Symfony charge toutes les routes à partir du fichier /app/config/routing.yml (et encore même ça on pourrait le customiser). Mais tu imagines s'il fallait lister toutes les routes de notre appli dans un seul fichier, il serait illisible. On importe donc d'autres fichiers en lui précisant qu'il doit les charger comme une ressource. Dans notre cas, il charge le fichier "@WCSCoavBundle/Resources/config/routing.yml" qui lui même charge le fichier "@WCSCoavBundle/Resources/config/routing/planemodel.yml" en préfixant toutes les routes par planemodel.

Mais alors elles viennent d'où toutes les routes qui s'affichent quand on lance la commande `php app/console debug:router` ?

Par défaut cette commande affiche les routes de l'environnement de dev. Essaie avec cette commande `php app/console debug:router --env=prod`. L'environnement de dev donne accès à un certain nombre de choses que tu ne veux pas forcément exposer aux utilisateurs finaux. C'est le cas de la barre en bas de l'écran par exemple.

Allons voir maintenant ce qu'a fait Symfony. Rends-toi sur l'URL suivante :

[http://localhost/flyaround/web/app\\_dev.php/planemodel](http://localhost/flyaround/web/app_dev.php/planemodel) Symfony a créé tout ce qu'il faut pour créer des avions, les lister, les mettre à jour et les supprimer. Tout ce qu'il faut c'est-à-dire :

- Un controller (PlaneModelController) avec les actions index, show, new, edit et delete
- Des vues (edit, index, new, show) dans le dossier `app/Resources/views/planemodel`
- Un formulaire et un fichier de tests (nous rentrerons dans ces détails plus tard)

1. Le routage (<http://jobeet.thuau.fr/le-routage>)

Tous les détails sur les routes de Symfony

2. La doc SF2 sur les routes (<http://symfony.com/doc/2.8/book/routing.html>)

La doc officielle montre les autres méthodes de définition des routes (XML, Annotations et PHP)

---

[Retour à la quête \(/quests/49\)](#)

[Valider \(/tasks/7174/validate?redirect=%2Fquests%2F49%\)](#)