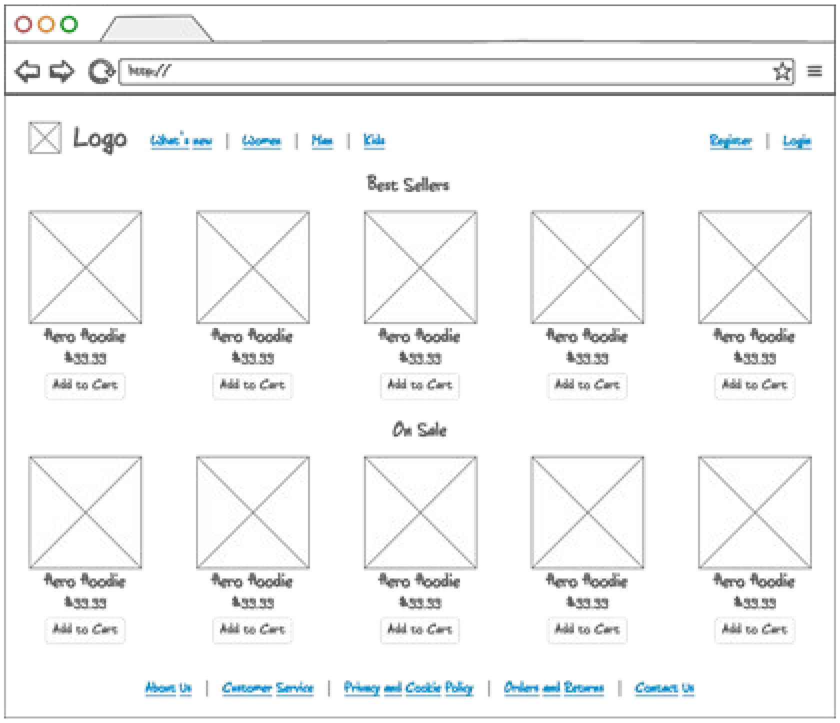
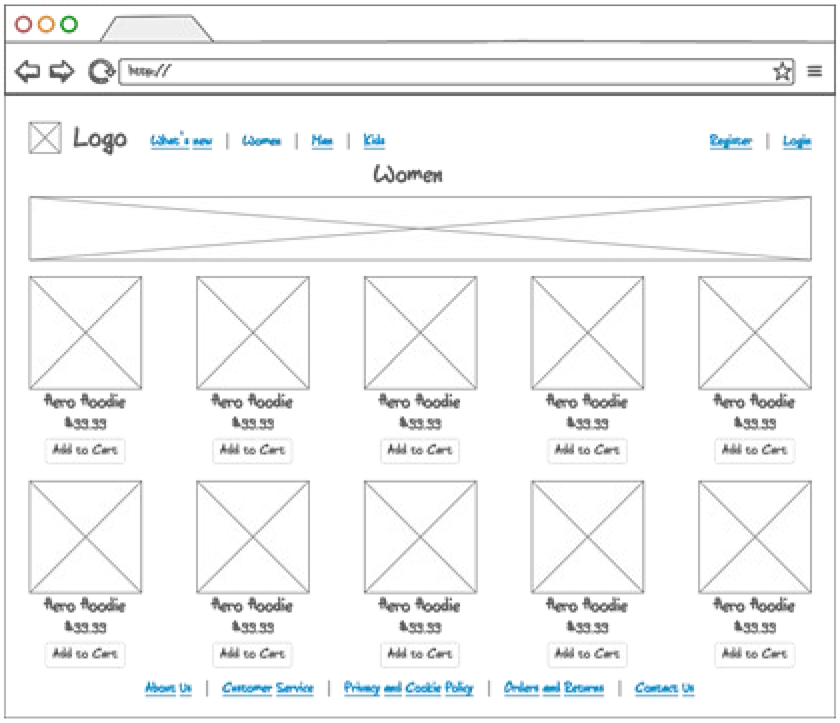
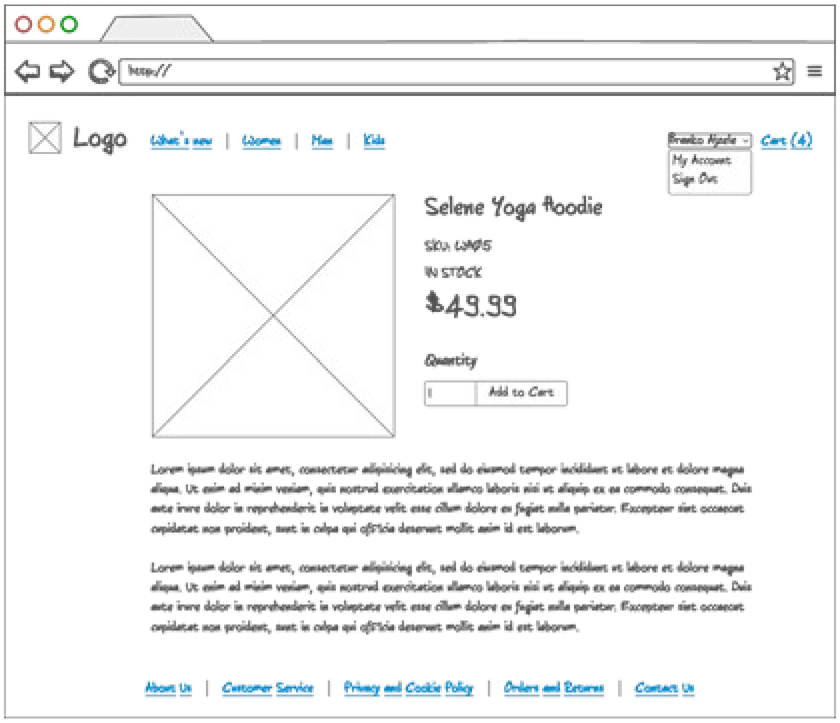
Chapter 4: Requirement Specification for a Modular Web Shop App




http://
Logo
What's new | Women | Men | Kids
Register | Login
Women
Aero Hoodie
$33.33
Add to Cart
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us


http://
Logo
What's new | Women | Men | Kids
Brooks Apple
My Account
Sign Out
Cart (4)
Add to Cart
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us

Selene Yoga Hoodie

SKU: WH05

IN STOCK

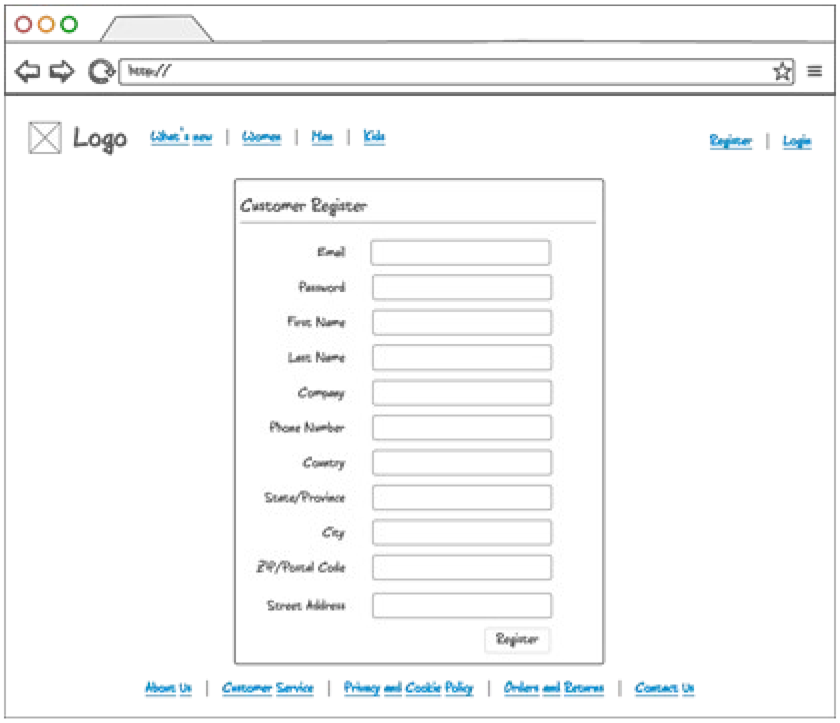
$49.99

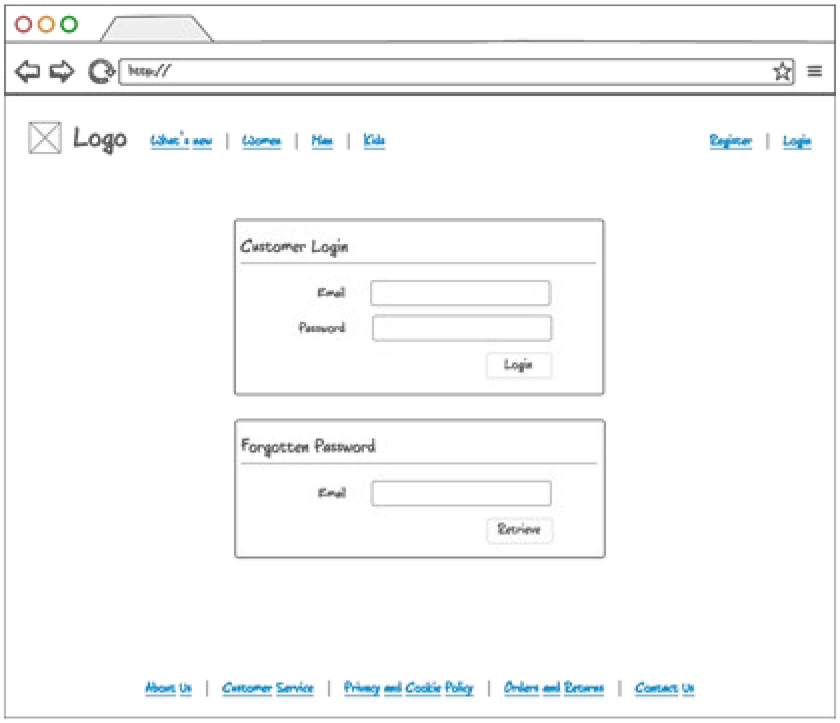
Quantity

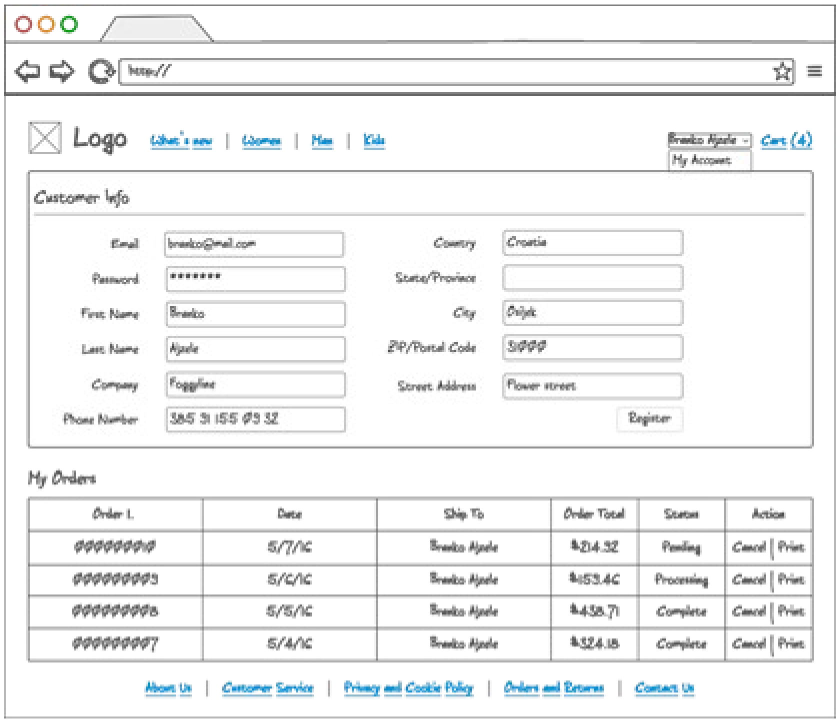
1

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.


http://
Logo
What's new
Women
Men
Kids
Register
Login
Customer Register
Email
Password
First Name
Last Name
Company
Phone Number
Country
State/Province
City
ZIP/Postal Code
Street Address
Register
About Us
Customer Service
Privacy and Cookie Policy
Orders and Returns
Contact Us


http://
Logo
What's new | Women | Men | Kids
Register | Login
Customer Login
Email
Password
Login
Forgotten Password
Email
Retrieve
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us



http://

Logo What's new | Women | Men | Kids

Brenko Ajzele Cart (4)

My Account

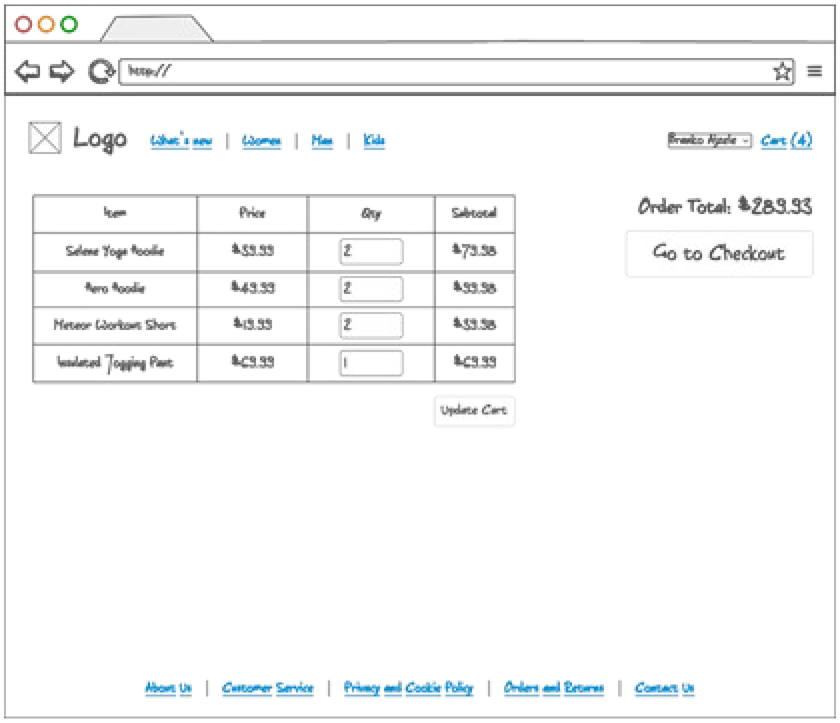
Customer Info

Email	brenko@mail.com	Country	Croatia
Password	********	State/Province	
First Name	Brenko	City	Osijek
Last Name	Ajzele	ZIP/Postal Code	31000
Company	Foggline	Street Address	Flower street
Phone Number	385 31 155 03 32		Register

My Orders

Order I.	Date	Ship To	Order Total	Status	Action
000000010	5/7/16	Brenko Ajzele	$214.32	Pending	Cancel | Print
000000009	5/6/16	Brenko Ajzele	$153.46	Processing	Cancel | Print
000000008	5/5/16	Brenko Ajzele	$438.71	Complete	Cancel | Print
000000007	5/4/16	Brenko Ajzele	$324.18	Complete	Cancel | Print

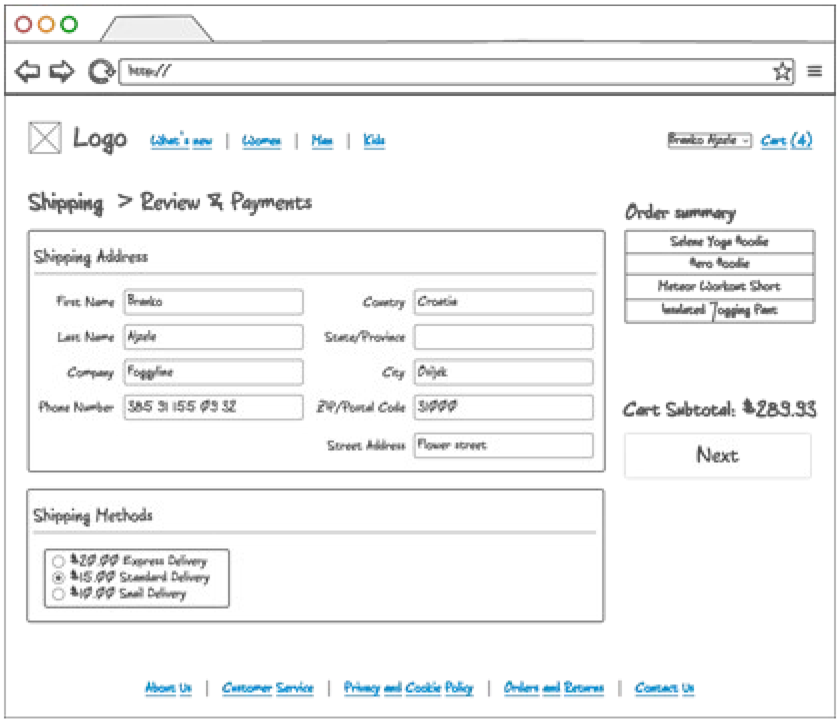
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us


http://
Logo
What's new | Women | Men | Kids
Branko Ajzele
Cart (4)

Item	Price	Qty	Subtotal
Selene Yoga Hoodie	$39.99	2	$79.98
Aero Hoodie	$49.99	2	$99.98
Meteor Workout Short	$19.99	2	$39.98
Isolated Jogging Pant	$69.99	1	$69.99

Order Total: $289.93

Go to Checkout
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us


http://
Logo
What's new | Women | Men | Kids
Branko Ajzele
Cart (4)
Next
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us

Shipping > Review & Payments

Shipping Address

First Name	Branko	Country	Croatia
Last Name	Ajzele	State/Province	
Company	Foggline	City	Osijek
Phone Number	385 31 155 03 32	ZIP/Postal Code	31000
		Street Address	Flower street

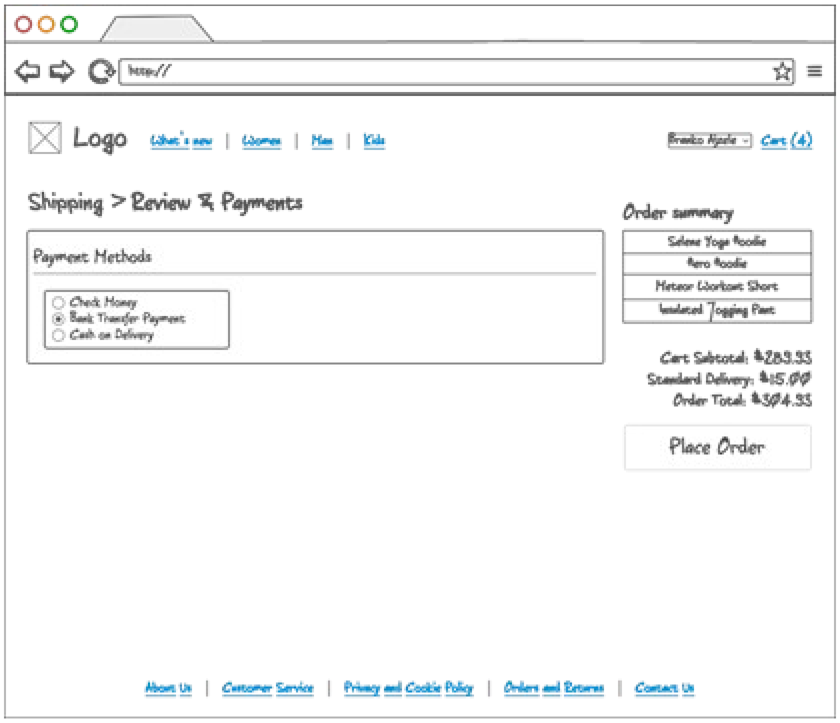
Shipping Methods

( ) $20.00 Express Delivery
[✓ $15.00 Standard Delivery]
( ) $10.00 Snail Delivery

Order summary

Selene Yoga Hoodie
Aero Hoodie
Meteor Workout Short
Isolated Jogging Pant

Cart Subtotal: $283.93


http://
Logo
What's new | Women | Men | Kids
Branko Ajzele
Cart (4)

Shipping > Review & Payments

Payment Methods

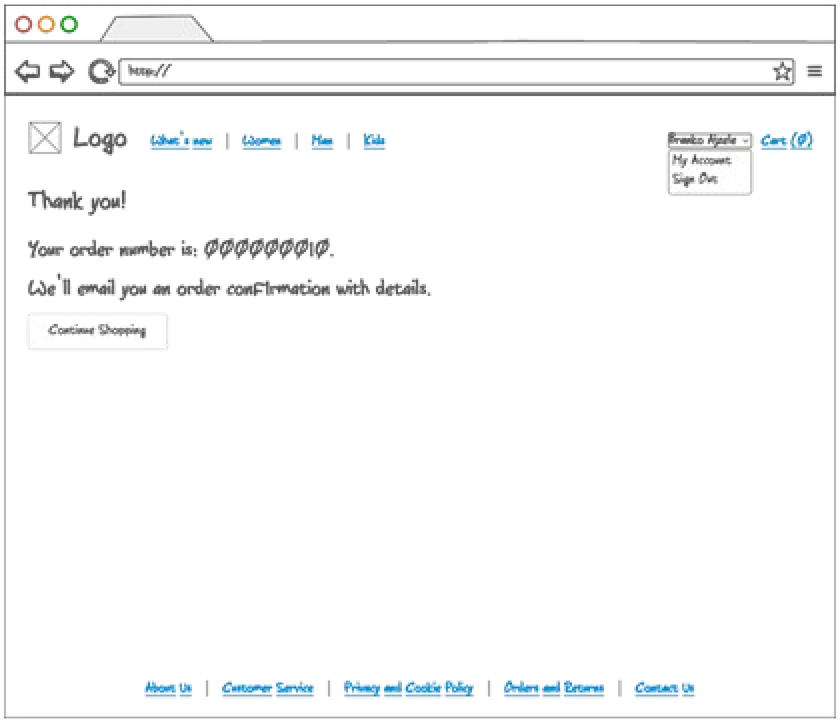
- ○ Check Money
- ● Bank Transfer Payment
- ○ Cash on Delivery

Order summary

Selene Yoga Hoodie
Aero Hoodie
Meteor Workout Short
Insulated Jogging Pant

Cart Subtotal: $283.33
Standard Delivery: $15.00
Order Total: $304.33

Place Order
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us

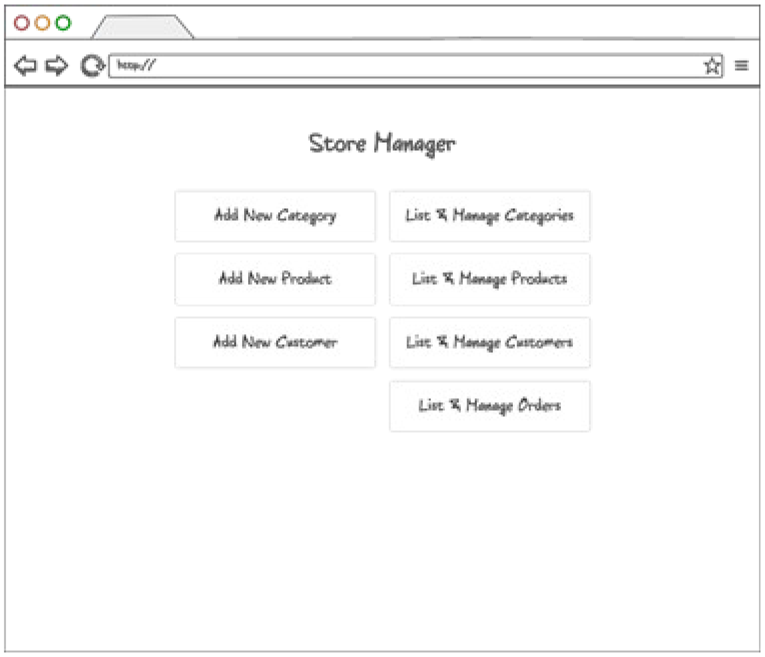

http://
Logo
What's new | Women | Men | Kids
Branko Ajzele
My Account
Sign Out
Cart (0)

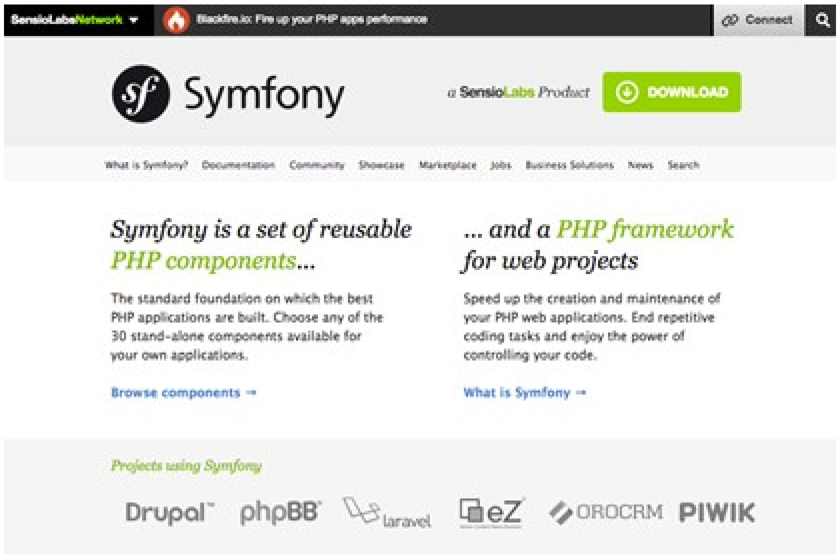
Thank you!

Your order number is: 000000010.

We'll email you an order confirmation with details.

Continue Shopping
About Us | Customer Service | Privacy and Cookie Policy | Orders and Returns | Contact Us




SensioLabsNetwork ▾ Blackfire.io: Fire up your PHP apps performance Connect

Symfony a SensioLabs Product DOWNLOAD

What is Symfony? Documentation Community Showcase Marketplace Jobs Business Solutions News Search

Symfony is a set of reusable PHP components...

The standard foundation on which the best PHP applications are built. Choose any of the 30 stand-alone components available for your own applications.

Browse components →

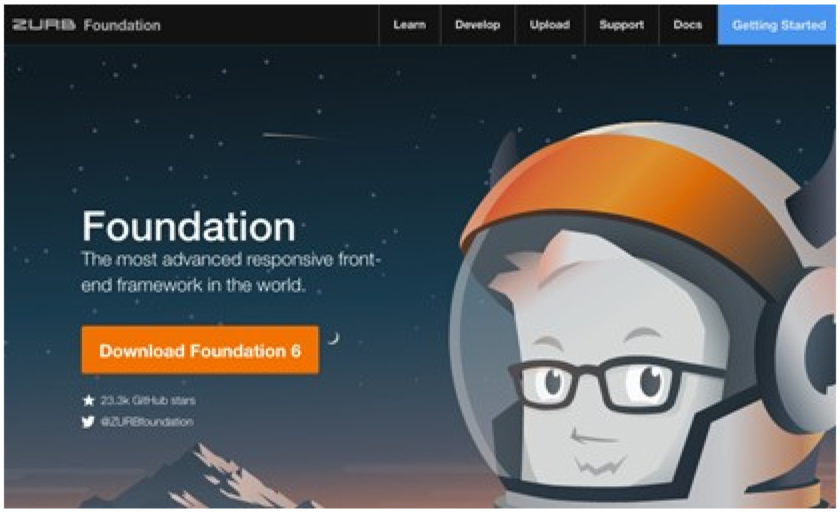
... and a PHP framework for web projects

Speed up the creation and maintenance of your PHP web applications. End repetitive coding tasks and enjoy the power of controlling your code.

What is Symfony →

Projects using Symfony

Drupal phpBB laravel eZ OROCRM PIWIK



Foundation

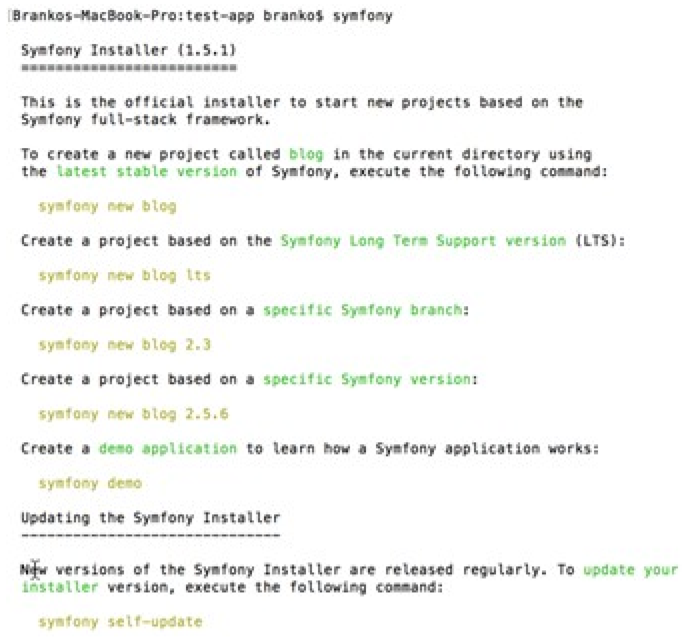
The most advanced responsive front-end framework in the world.

Download Foundation 6

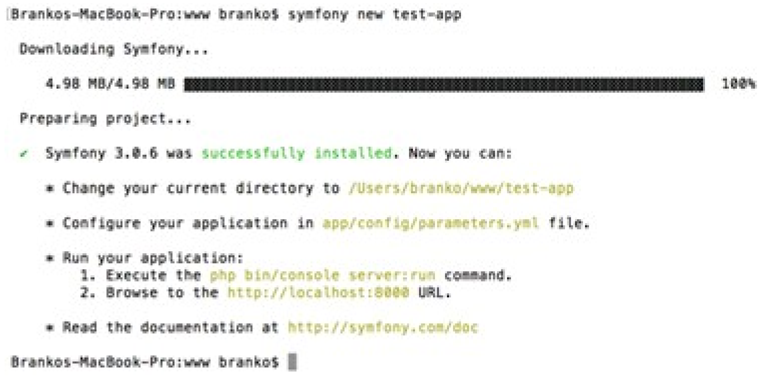
23.3k GitHub stars

@ZURBfoundation

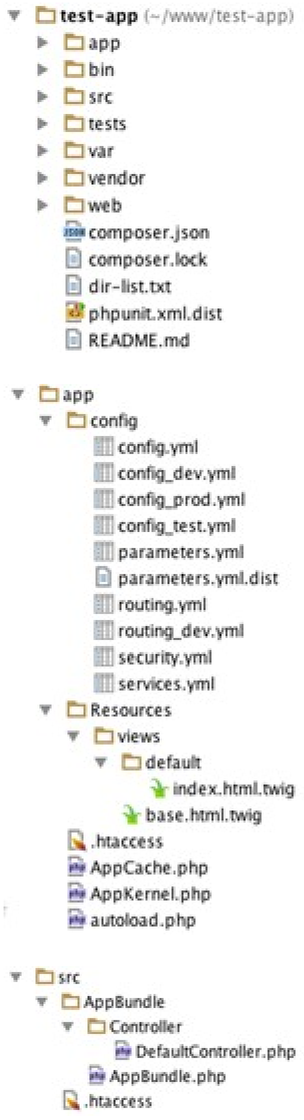
Chapter 5: Symfony at a Glance



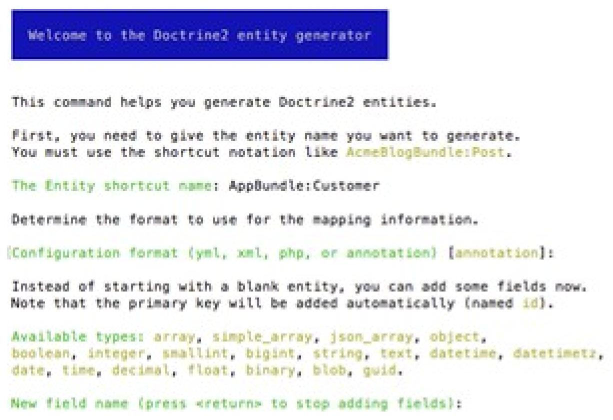
```
Brankos-MacBook-Pro:test-app branko$ symfony

 Symfony Installer (1.5.1)
 =========================

 This is the official installer to start new projects based on the
 Symfony full-stack framework.

 To create a new project called blog in the current directory using
 the latest stable version of Symfony, execute the following command:

   symfony new blog

 Create a project based on the Symfony Long Term Support version (LTS):

   symfony new blog lts

 Create a project based on a specific Symfony branch:

   symfony new blog 2.3

 Create a project based on a specific Symfony version:

   symfony new blog 2.5.6

 Create a demo application to learn how a Symfony application works:

   symfony demo

 Updating the Symfony Installer
 ------------------------------

 New versions of the Symfony Installer are released regularly. To update your
 installer version, execute the following command:

   symfony self-update
```



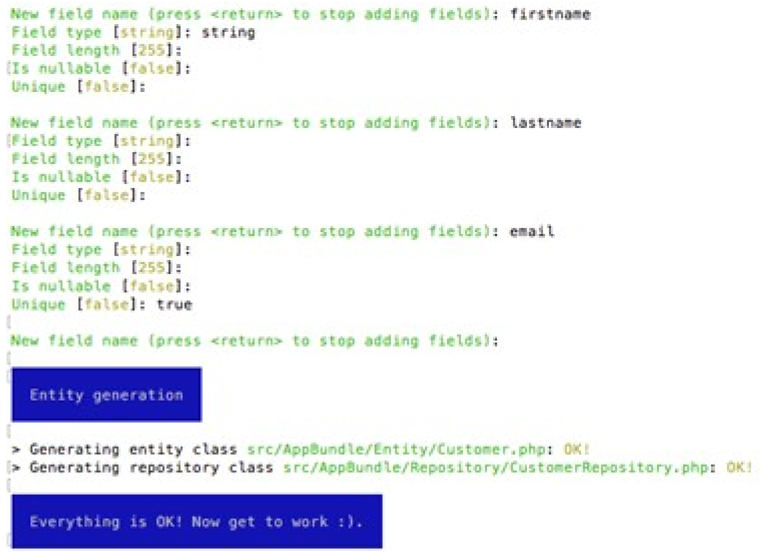
```
Brankos-MacBook-Pro:www branko$ symfony new test-app

 Downloading Symfony...

    4.98 MB/4.98 MB ████████████████████████████████████████████████████████  100%

 Preparing project...

 ✓  Symfony 3.0.6 was successfully installed. Now you can:

    * Change your current directory to /Users/branko/www/test-app

    * Configure your application in app/config/parameters.yml file.

    * Run your application:
        1. Execute the php bin/console server:run command.
        2. Browse to the http://localhost:8000 URL.

    * Read the documentation at http://symfony.com/doc

Brankos-MacBook-Pro:www branko$ 
```



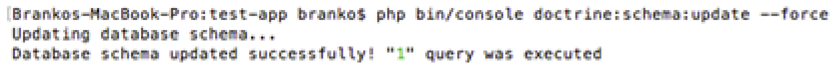
```
▼ test-app (~/www/test-app)
  ▶ app
  ▶ bin
  ▶ src
  ▶ tests
  ▶ var
  ▶ vendor
  ▶ web
    composer.json
    composer.lock
    dir-list.txt
    phpunit.xml.dist
    README.md

▼ app
  ▼ config
      config.yml
      config_dev.yml
      config_prod.yml
      config_test.yml
      parameters.yml
      parameters.yml.dist
      routing.yml
      routing_dev.yml
      security.yml
      services.yml
  ▼ Resources
    ▼ views
      ▼ default
          index.html.twig
        base.html.twig
    .htaccess
    AppCache.php
    AppKernel.php
    autoload.php

▼ src
  ▼ AppBundle
    ▼ Controller
        DefaultController.php
      AppBundle.php
    .htaccess
```



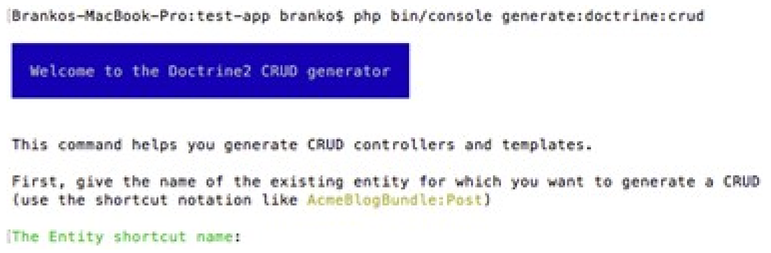
```
  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: AppBundle:Customer

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields):
```



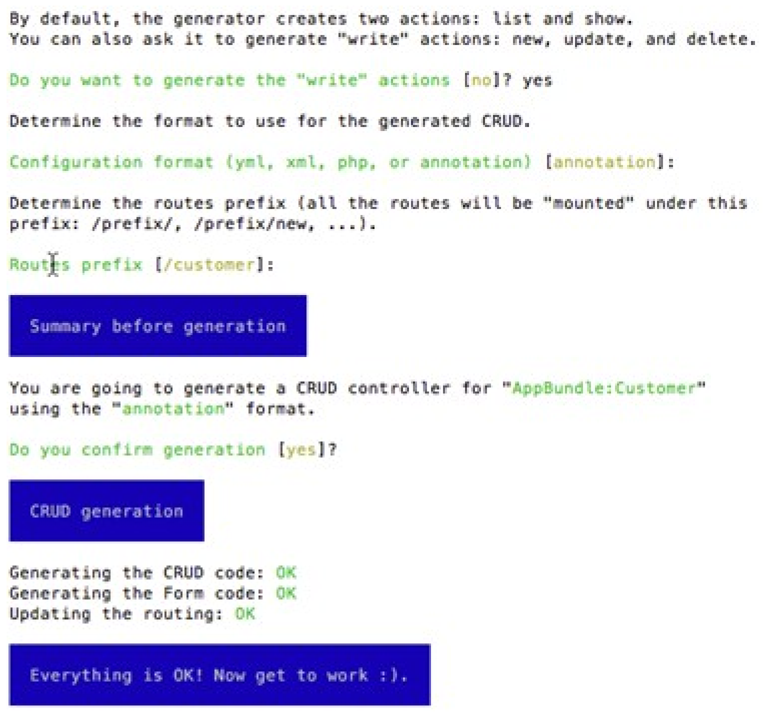
```
New field name (press <return> to stop adding fields): firstname
Field type [string]: string
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): lastname
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): email
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]: true

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class src/AppBundle/Entity/Customer.php: OK!
> Generating repository class src/AppBundle/Repository/CustomerRepository.php: OK!

  Everything is OK! Now get to work :).
```



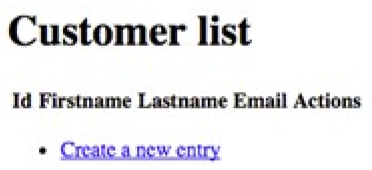
```
Brankos-MacBook-Pro:test-app branko$ php bin/console doctrine:schema:update --force
Updating database schema...
Database schema updated successfully! "1" query was executed
```



```
Brankos-MacBook-Pro:test-app branko$ php bin/console generate:doctrine:crud

  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:
```



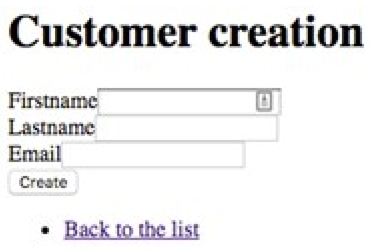
```
By default, the generator creates two actions: list and show.
You can also ask it to generate "write" actions: new, update, and delete.

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

Determine the format to use for the generated CRUD.

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

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

Routes prefix [/customer]:

  Summary before generation

You are going to generate a CRUD controller for "AppBundle:Customer"
using the "annotation" format.

Do you confirm generation [yes]?

  CRUD generation

Generating the CRUD code: OK
Generating the Form code: OK
Updating the routing: OK

  Everything is OK! Now get to work :).
```



Customer list

Id Firstname Lastname Email Actions

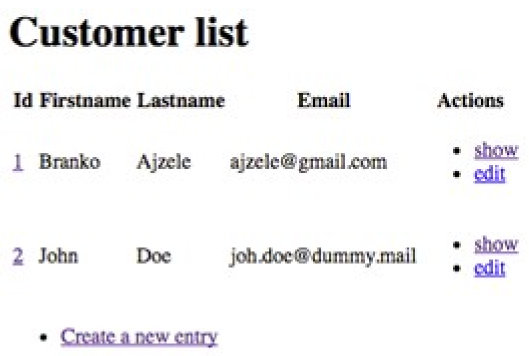
- Create a new entry



Customer creation

Firstname
Lastname
Email
Create

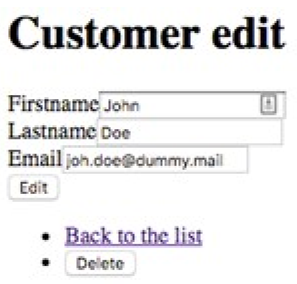
- Back to the list



Customer list

Id	Firstname	Lastname	Email	Actions
1	Branko	Ajzele	ajzele@gmail.com	• show • edit
2	John	Doe	joh.doe@dummy.mail	• show • edit

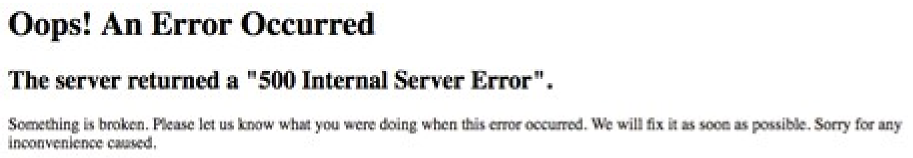
- Create a new entry



Customer edit

Firstname John
Lastname Doe
Email joh.doe@dummy.mail

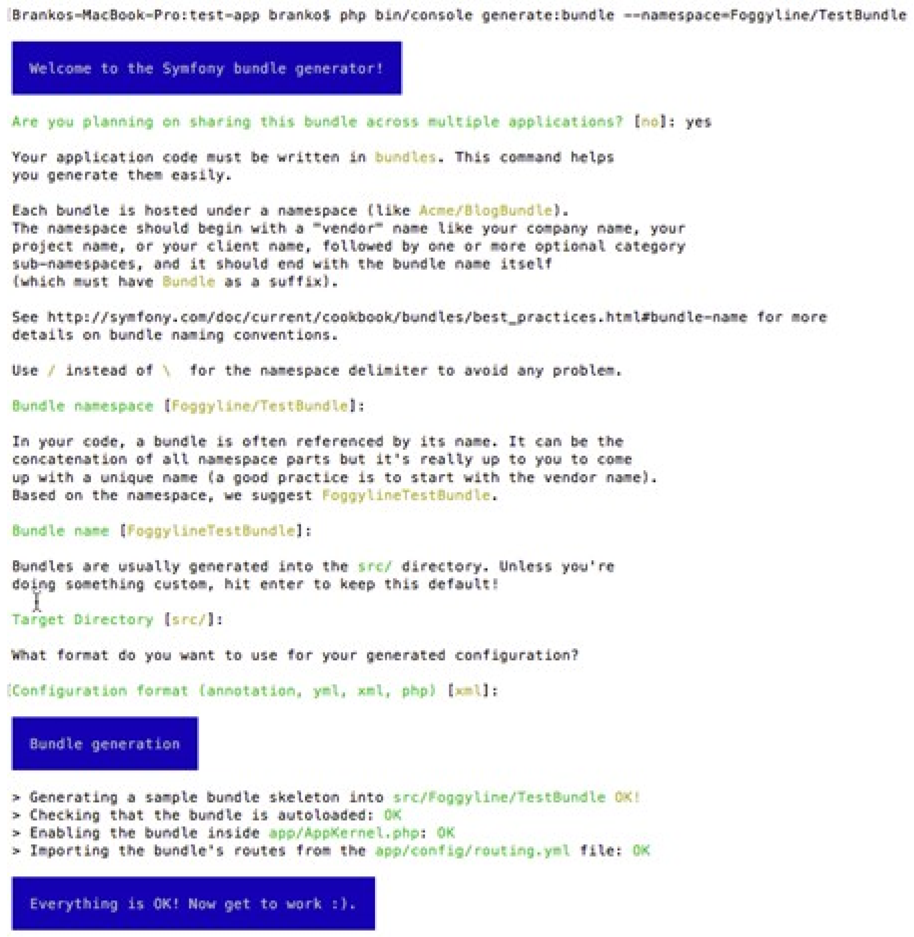
- Back to the list
-



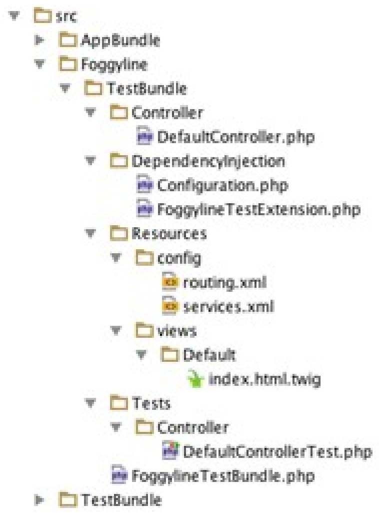
Oops! An Error Occurred

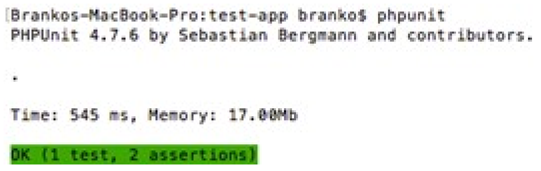
The server returned a "500 Internal Server Error".

Something is broken. Please let us know what you were doing when this error occurred. We will fix it as soon as possible. Sorry for any inconvenience caused.

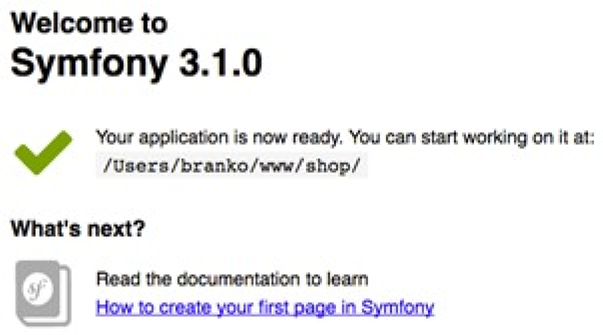


```
Brankos-MacBook-Pro:test-app branko$ php bin/console generate:bundle --namespace=Foggyline/TestBundle

  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/TestBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineTestBundle.

Bundle name [FoggylineTestBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/TestBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```


```
▼ src
  ▶ AppBundle
  ▼ Foggyline
    ▼ TestBundle
      ▼ Controller
          DefaultController.php
      ▼ DependencyInjection
          Configuration.php
          FoggylineTestExtension.php
      ▼ Resources
        ▼ config
            routing.xml
            services.xml
        ▼ views
          ▼ Default
              index.html.twig
      ▼ Tests
        ▼ Controller
            DefaultControllerTest.php
        FoggylineTestBundle.php
  ▶ TestBundle
```


```
Brankos-MacBook-Pro:test-app branko$ phpunit
PHPUnit 4.7.6 by Sebastian Bergmann and contributors.

.

Time: 545 ms, Memory: 17.00Mb

OK (1 test, 2 assertions)
```

Chapter 6: Building the Core Module

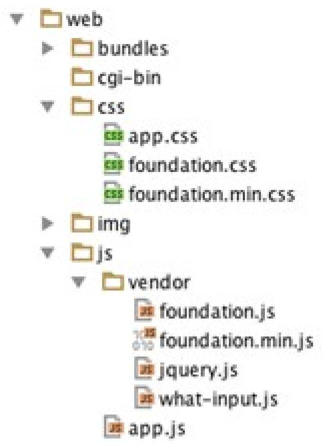


Welcome to
Symfony 3.1.0

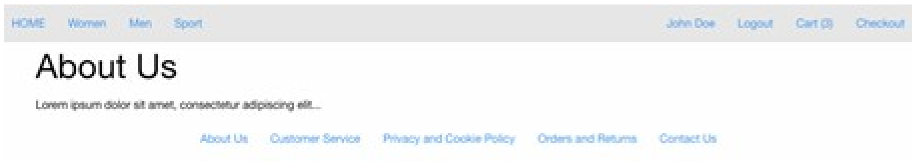
Your application is now ready. You can start working on it at:
`/Users/branko/www/shop/`

What's next?

Read the documentation to learn
How to create your first page in Symfony

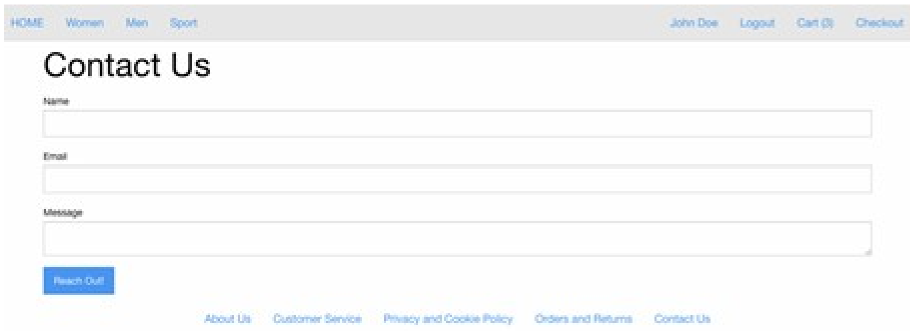


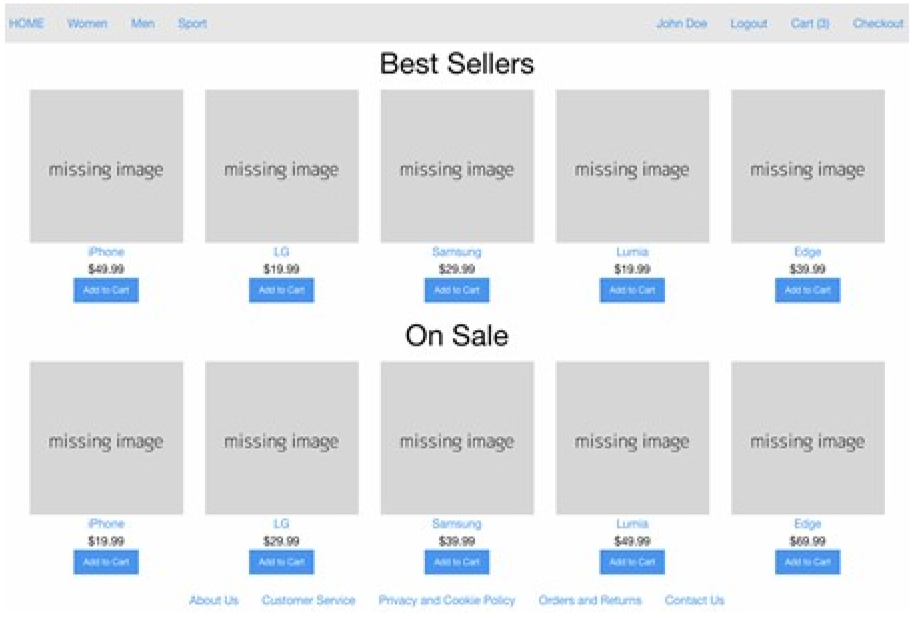
```
web
  bundles
  cgi-bin
  css
    app.css
    foundation.css
    foundation.min.css
  img
  js
    vendor
      foundation.js
      foundation.min.js
      jquery.js
      what-input.js
    app.js
```

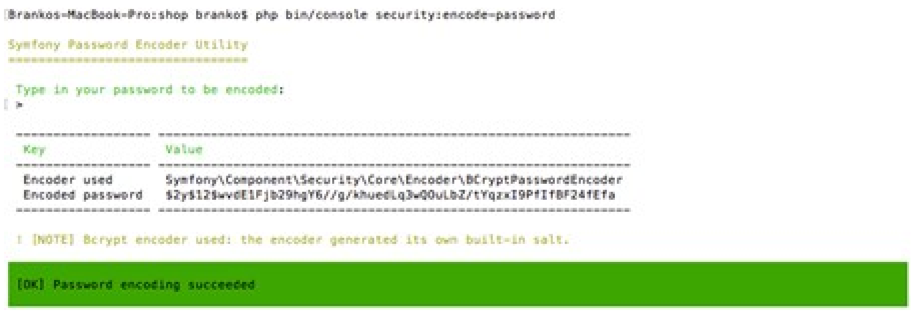


About Us

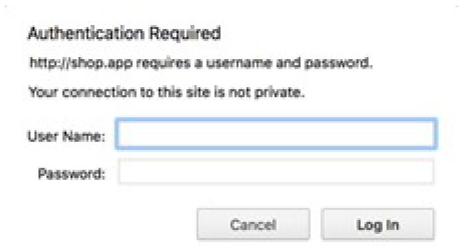
Lorem ipsum dolor sit amet, consectetur adipiscing elit...







```
Brankos-MacBook-Pro:shop branko$ php bin/console security:encode-password

Symfony Password Encoder Utility
================================

 Type in your password to be encoded:
 >

 ------------------ -----------------------------------------------------------------
  Key                Value
 ------------------ -----------------------------------------------------------------
  Encoder used       Symfony\Component\Security\Core\Encoder\BCryptPasswordEncoder
  Encoded password   $2y$12$wvdE1Fjb29hgY6//g/khuedLq3wQOuLbZ/tYqzxI9PfIfBF24fEfa
 ------------------ -----------------------------------------------------------------

 ! [NOTE] Bcrypt encoder used: the encoder generated its own built-in salt.

 [OK] Password encoding succeeded
```



Authentication Required

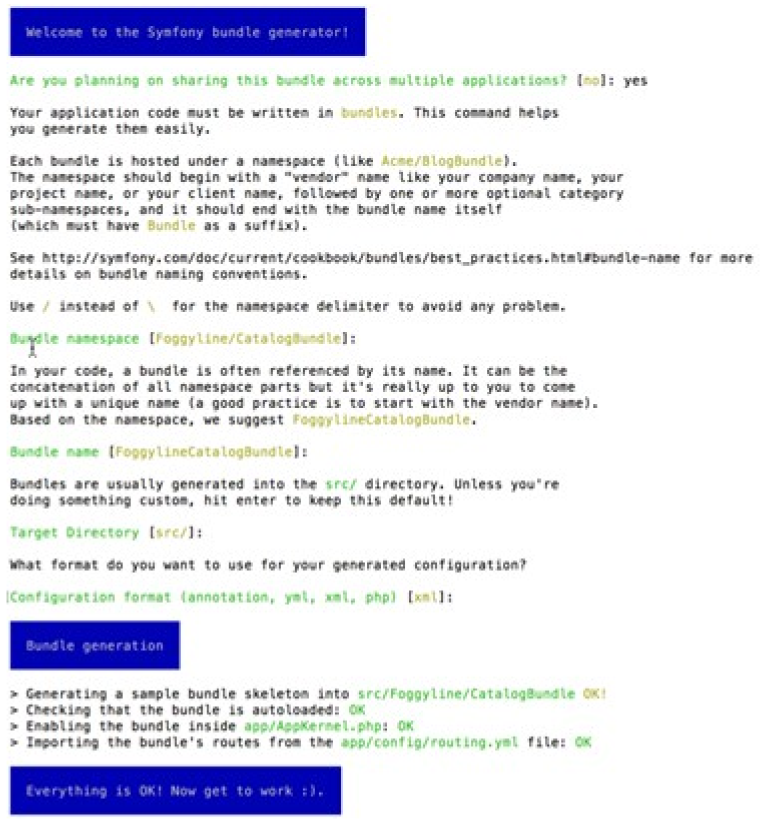
http://shop.app requires a username and password.

Your connection to this site is not private.

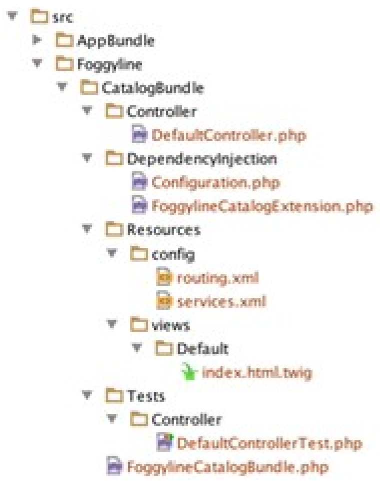
User Name:

Password:

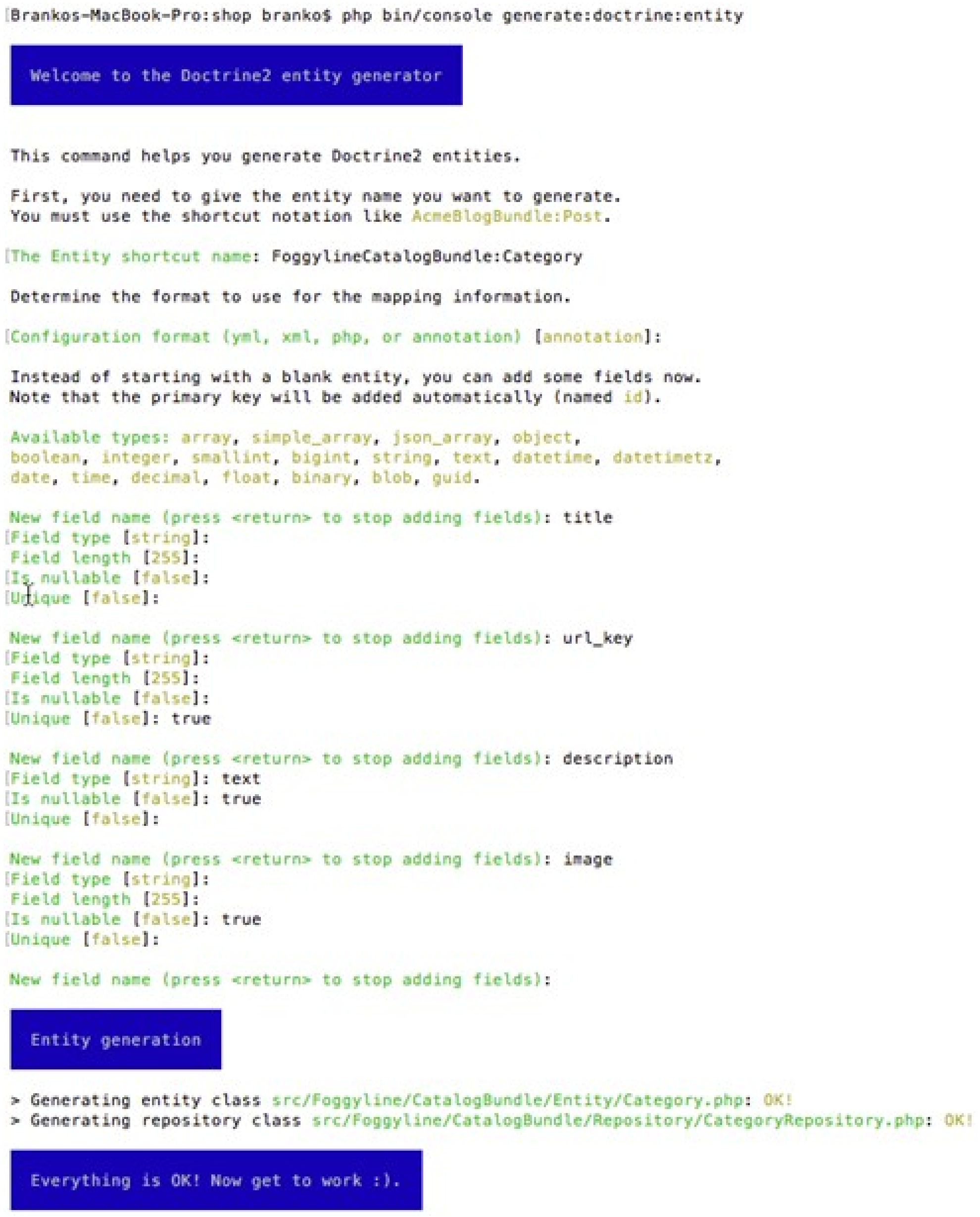
Chapter 7: Building the Catalog Module



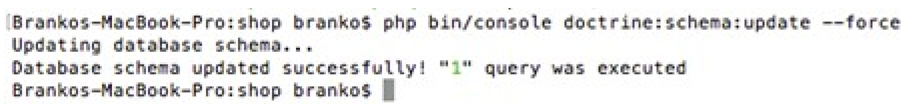
```
  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/CatalogBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineCatalogBundle.

Bundle name [FoggylineCatalogBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/CatalogBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```



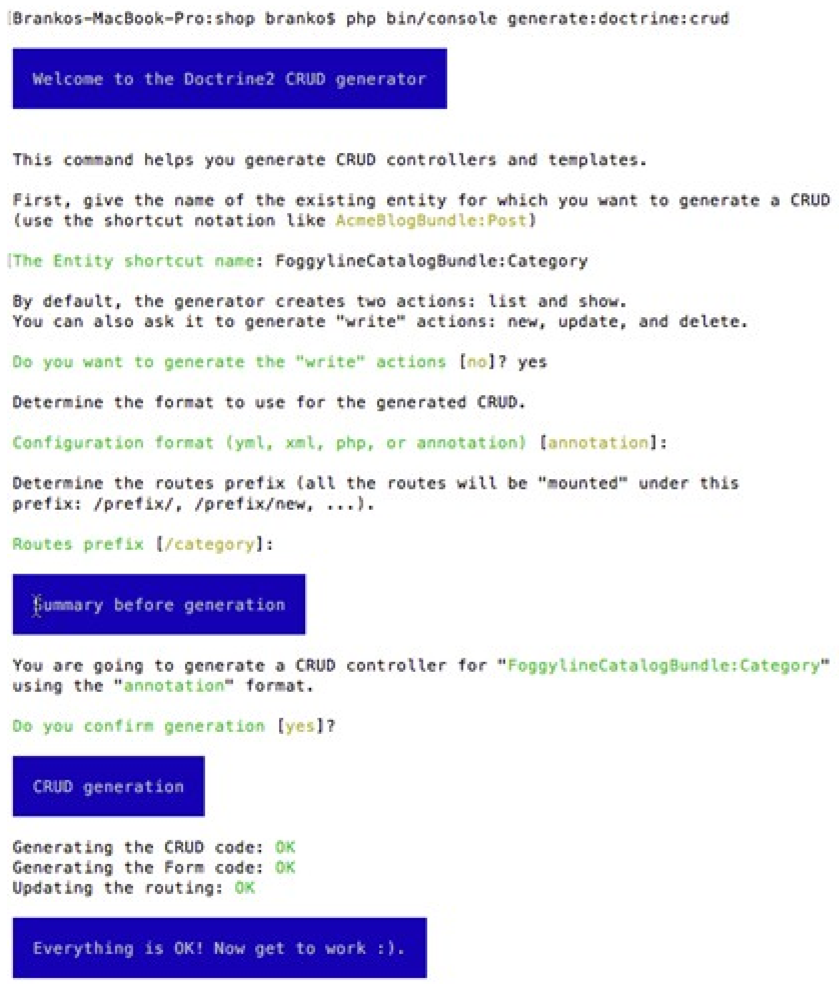
```
▼ src
  ▶ AppBundle
  ▼ Foggyline
    ▼ CatalogBundle
      ▼ Controller
          DefaultController.php
      ▼ DependencyInjection
          Configuration.php
          FoggylineCatalogExtension.php
      ▼ Resources
        ▼ config
            routing.xml
            services.xml
        ▼ views
          ▼ Default
              index.html.twig
      ▼ Tests
        ▼ Controller
            DefaultControllerTest.php
        FoggylineCatalogBundle.php
```



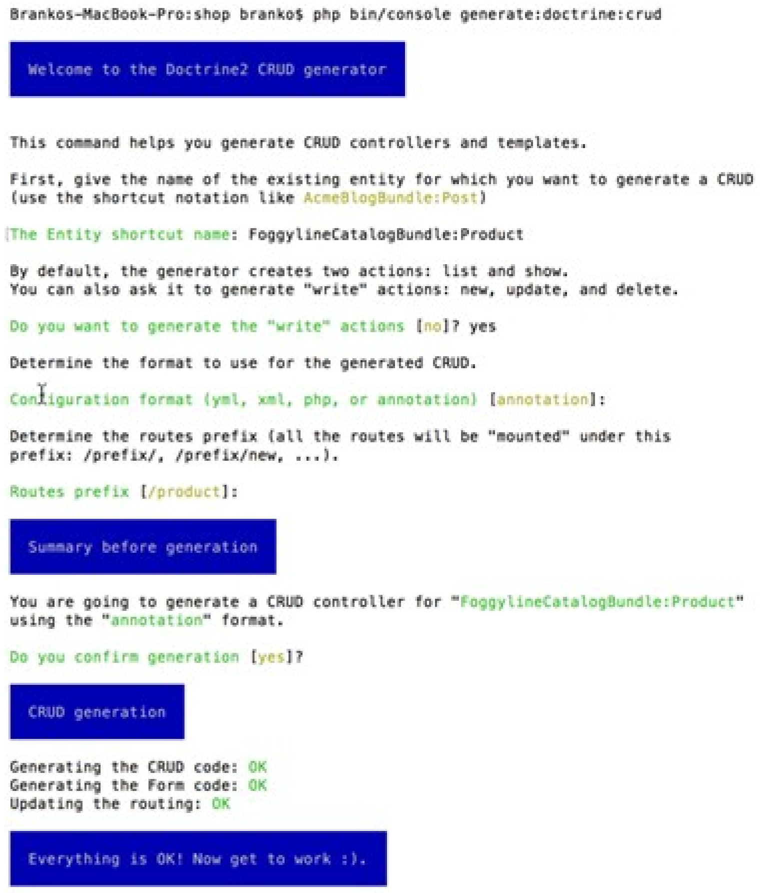
```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineCatalogBundle:Category

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): title
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): url_key
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]: true

New field name (press <return> to stop adding fields): description
Field type [string]: text
Is nullable [false]: true
Unique [false]:

New field name (press <return> to stop adding fields): image
Field type [string]:
Field length [255]:
Is nullable [false]: true
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class src/Foggyline/CatalogBundle/Entity/Category.php: OK!
> Generating repository class src/Foggyline/CatalogBundle/Repository/CategoryRepository.php: OK!

  Everything is OK! Now get to work :).
```



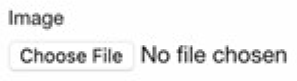
```
Brankos-MacBook-Pro:shop branko$ php bin/console doctrine:schema:update --force
Updating database schema...
Database schema updated successfully! "1" query was executed
Brankos-MacBook-Pro:shop branko$
```

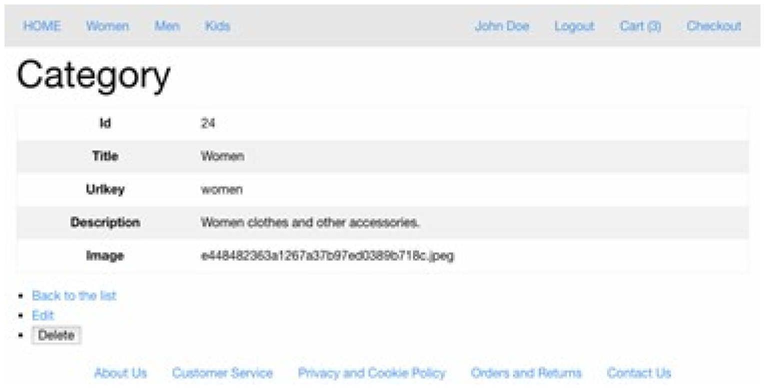


```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:doctrine:crud

  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: FoggylineCatalogBundle:Category

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 (yml, xml, php, or annotation) [annotation]:

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

Routes prefix [/category]:

  Summary before generation

You are going to generate a CRUD controller for "FoggylineCatalogBundle:Category"
using the "annotation" format.

Do you confirm generation [yes]?

  CRUD generation

Generating the CRUD code: OK
Generating the Form code: OK
Updating the routing: OK

  Everything is OK! Now get to work :).
```



```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:doctrine:crud

  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: FoggylineCatalogBundle:Product

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 (yml, xml, php, or annotation) [annotation]:

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

Routes prefix [/product]:

  Summary before generation

You are going to generate a CRUD controller for "FoggylineCatalogBundle:Product"
using the "annotation" format.

Do you confirm generation [yes]?

  CRUD generation

Generating the CRUD code: OK
Generating the Form code: OK
Updating the routing: OK

  Everything is OK! Now get to work :).
```

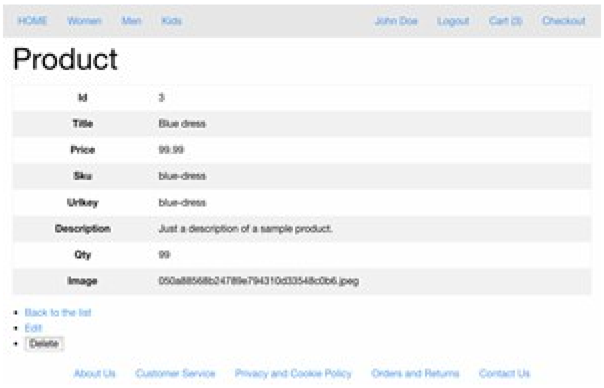




Category

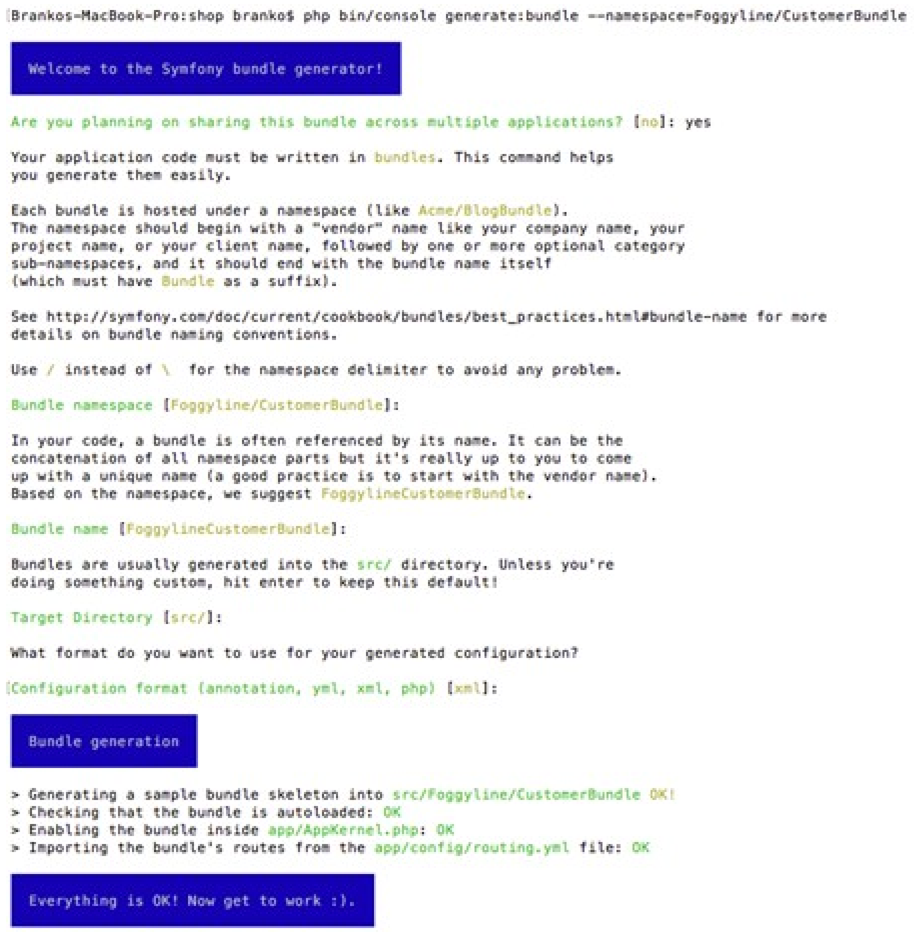
Id	24
Title	Women
Urlkey	women
Description	Women clothes and other accessories.
Image	e448482363a1267a37b97ed0389b718c.jpeg

- Back to the list
- Edit
- Delete

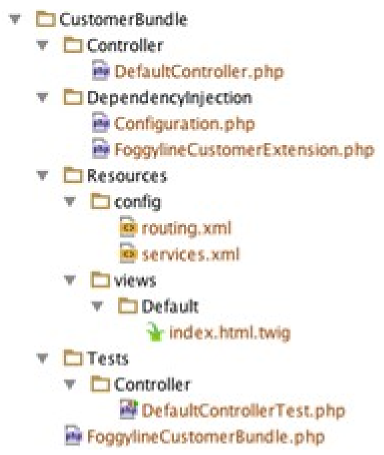


Product

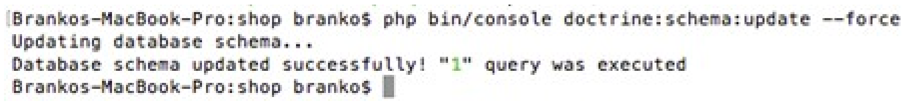
Id	3
Title	Blue dress
Price	99.99
Sku	blue-dress
Urlkey	blue-dress
Description	Just a description of a sample product.
Qty	99
Image	050a88568b24789e794310d33548c0b6.jpeg



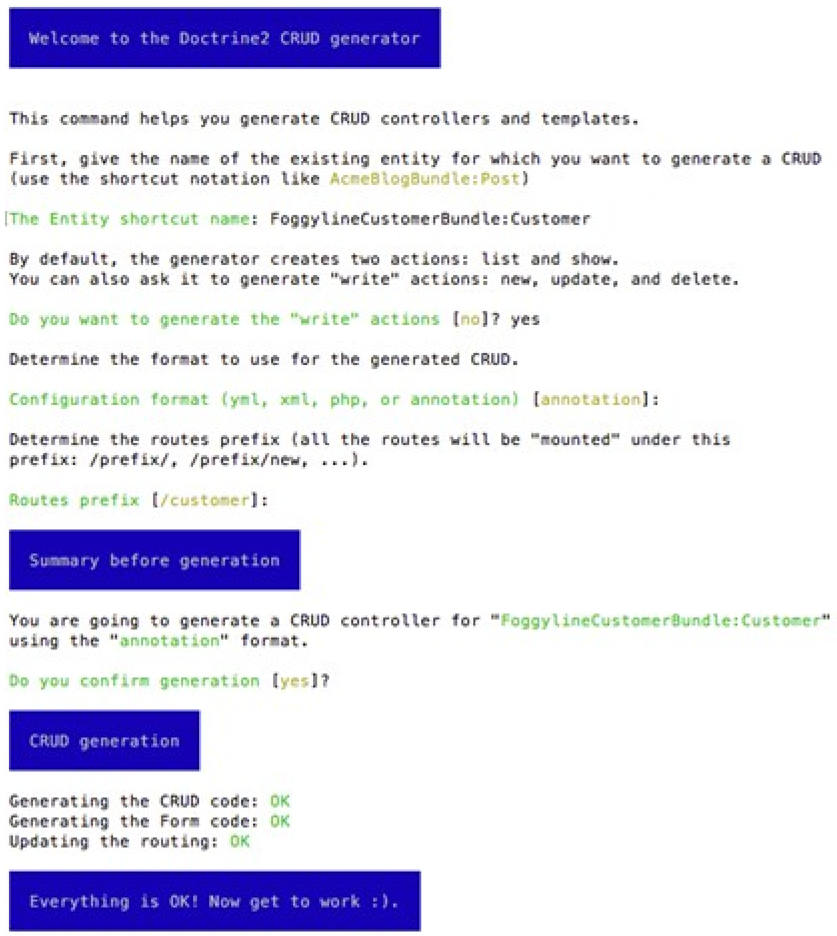
```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:bundle --namespace=Foggyline/CustomerBundle

  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/CustomerBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineCustomerBundle.

Bundle name [FoggylineCustomerBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/CustomerBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```



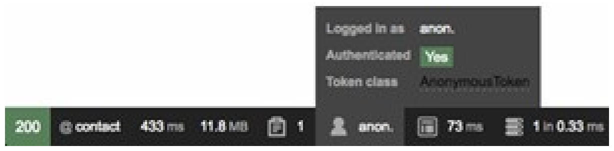
```
CustomerBundle
    Controller
        DefaultController.php
    DependencyInjection
        Configuration.php
        FoggylineCustomerExtension.php
    Resources
        config
            routing.xml
            services.xml
        views
            Default
                index.html.twig
    Tests
        Controller
            DefaultControllerTest.php
    FoggylineCustomerBundle.php
```



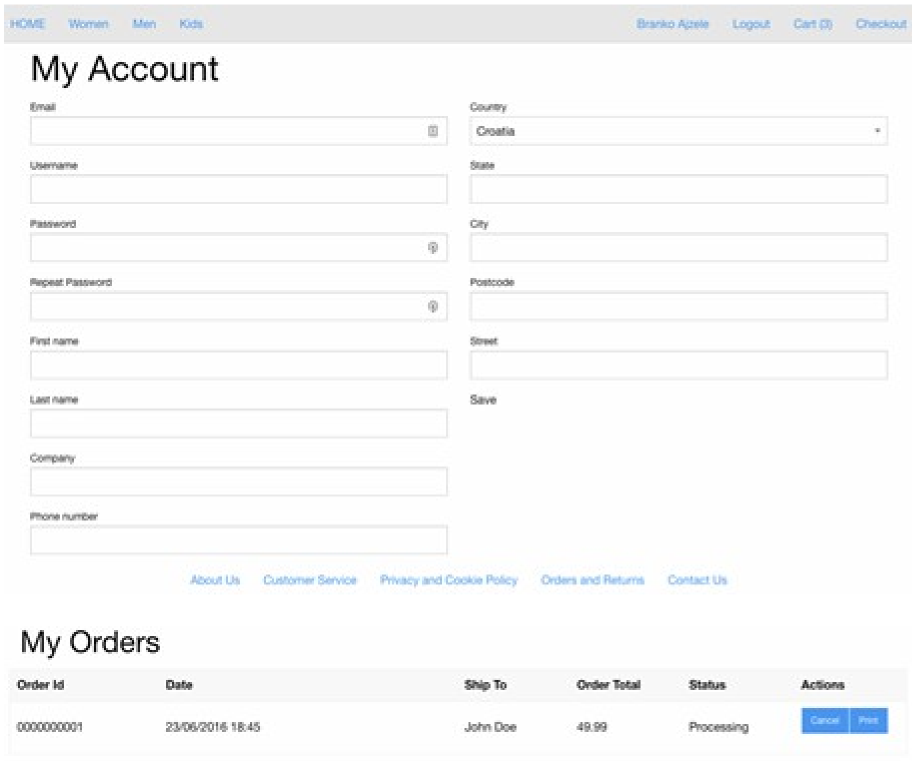
```
Brankos-MacBook-Pro:shop branko$ php bin/console doctrine:schema:update --force
Updating database schema...
Database schema updated successfully! "1" query was executed
Brankos-MacBook-Pro:shop branko$
```



```
  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: FoggylineCustomerBundle:Customer

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 (yml, xml, php, or annotation) [annotation]:

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

Routes prefix [/customer]:

  Summary before generation

You are going to generate a CRUD controller for "FoggylineCustomerBundle:Customer"
using the "annotation" format.

Do you confirm generation [yes]?

  CRUD generation

Generating the CRUD code: OK
Generating the Form code: OK
Updating the routing: OK

  Everything is OK! Now get to work :).
```



```
Logged in as    anon.
Authenticated   Yes
Token class     AnonymousToken

200  @contact  433 ms  11.8 MB  1  anon.  73 ms  1 in 0.33 ms
```


HOME Women Men Kids Branko Ajzele Logout Cart (3) Checkout

My Account

Email

Username

Password

Repeat Password

First name

Last name

Company

Phone number

Country
Croatia

State

City

Postcode

Street

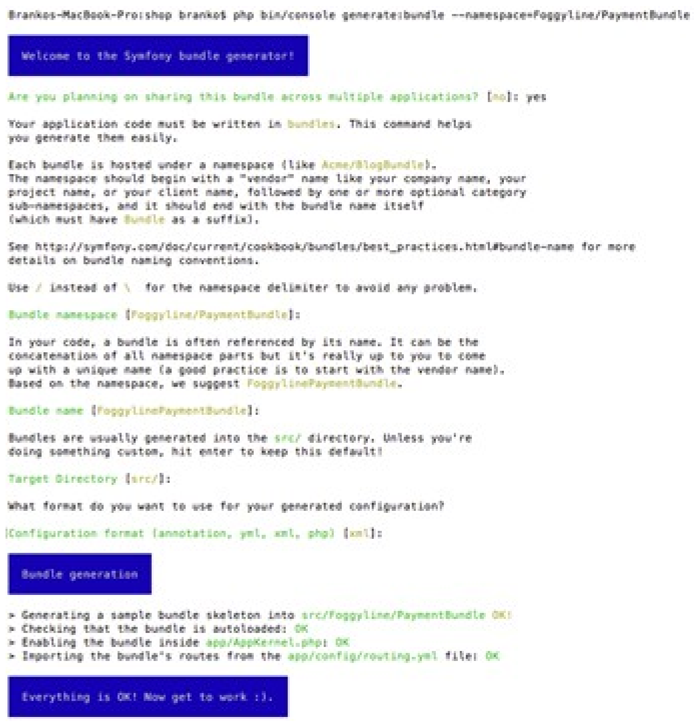
Save

About Us Customer Service Privacy and Cookie Policy Orders and Returns Contact Us

My Orders

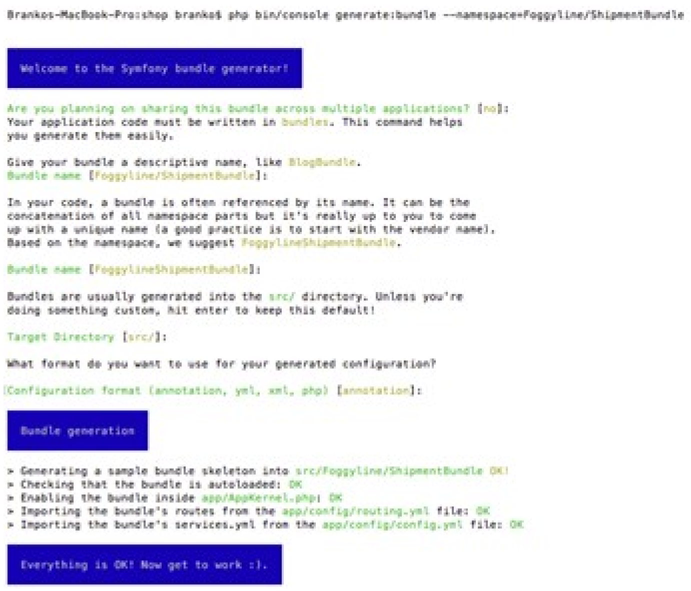
Order Id	Date	Ship To	Order Total	Status	Actions
0000000001	23/06/2016 18:45	John Doe	49.99	Processing	Cancel Print

Chapter 9: Building the Payment Module



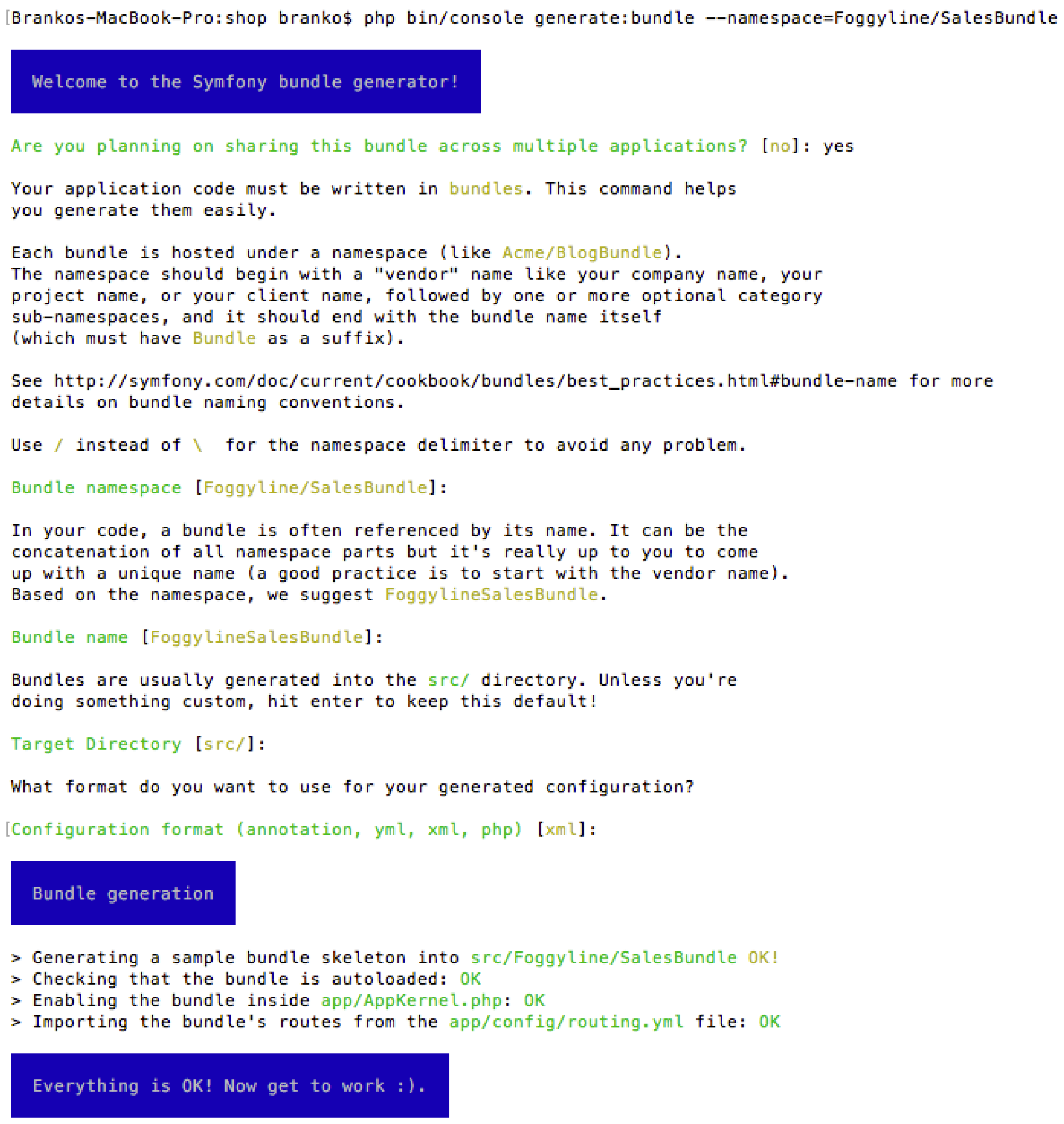
```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:bundle --namespace=Foggyline/PaymentBundle

  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/PaymentBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylinePaymentBundle.

Bundle name [FoggylinePaymentBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/PaymentBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```

Chapter 10: Building the Shipment Module

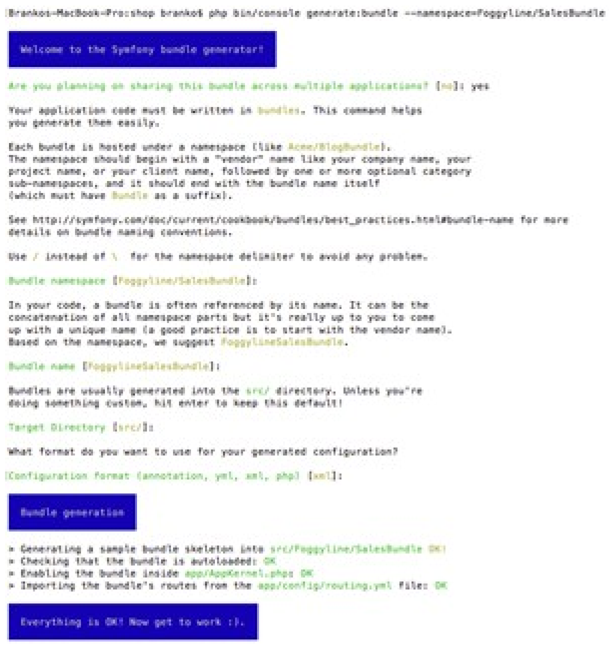


```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:bundle --namespace=Foggyline/ShipmentBundle


  Welcome to the Symfony bundle generator!


Are you planning on sharing this bundle across multiple applications? [no]:
Your application code must be written in bundles. This command helps
you generate them easily.

Give your bundle a descriptive name, like BlogBundle.
Bundle name [Foggyline/ShipmentBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineShipmentBundle.

Bundle name [FoggylineShipmentBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [annotation]:


  Bundle generation


> Generating a sample bundle skeleton into src/Foggyline/ShipmentBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK
> Importing the bundle's services.yml from the app/config/config.yml file: OK


  Everything is OK! Now get to work :).
```

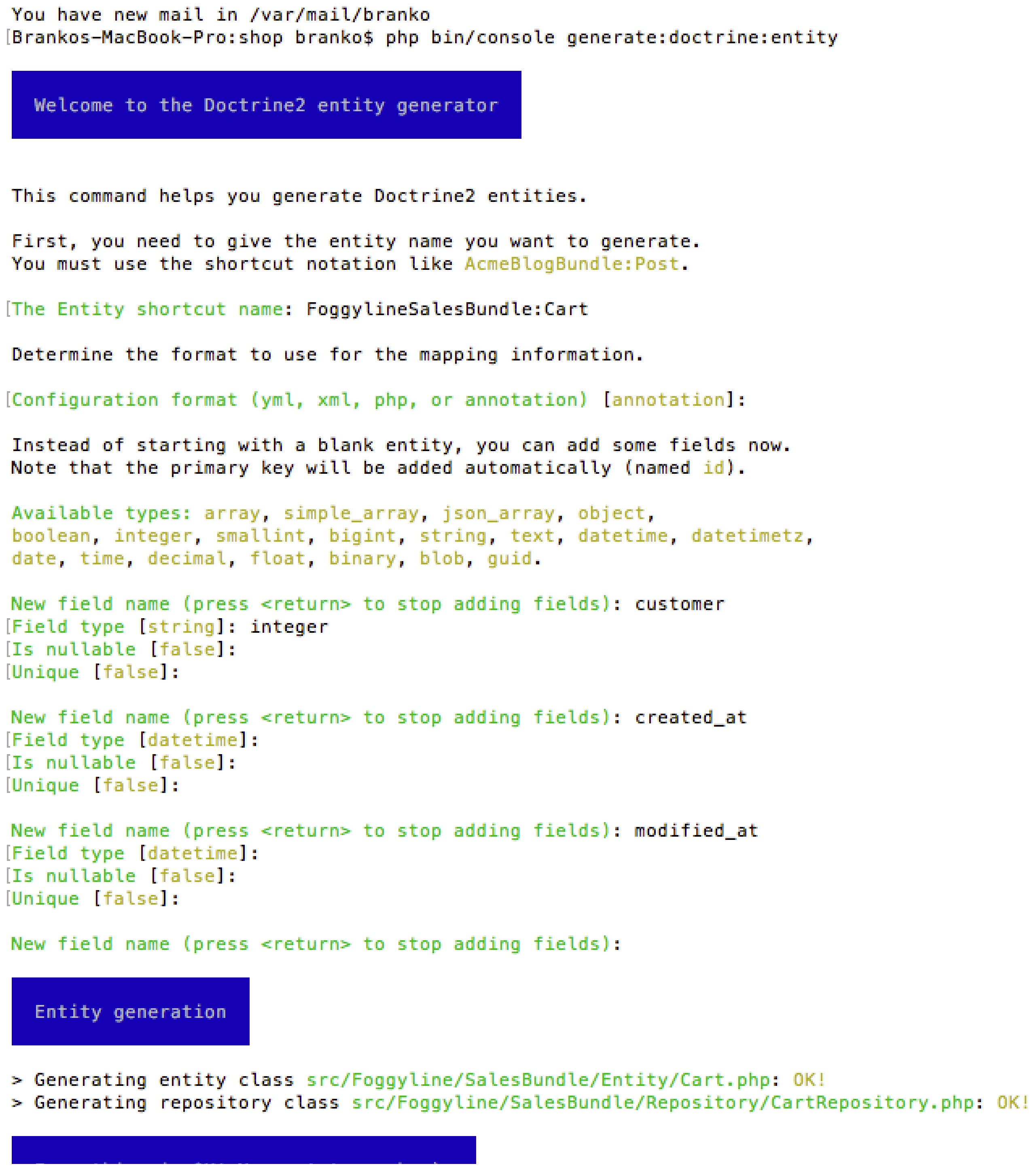
Chapter 11: Building the Sales Module



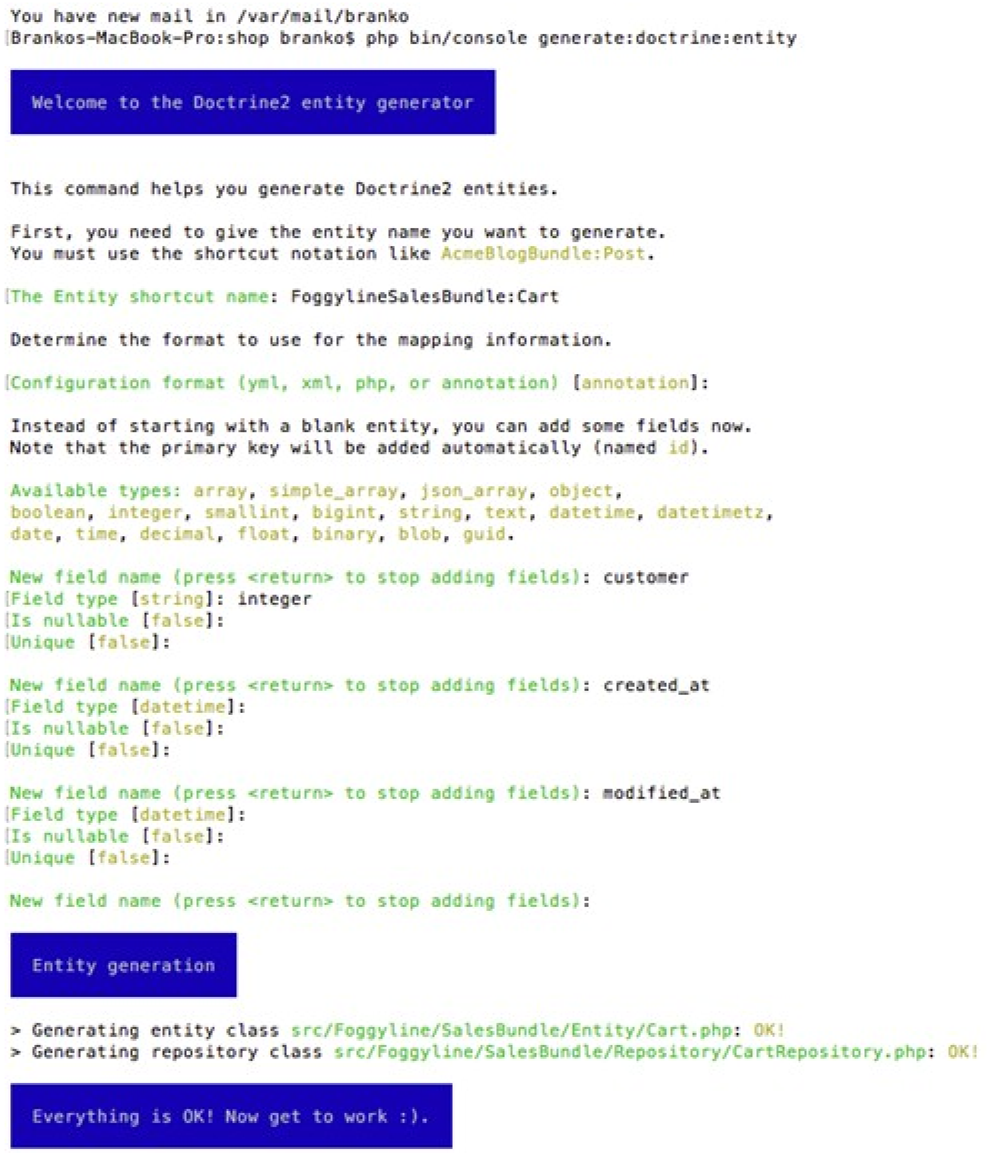
```
[Brankos-MacBook-Pro:shop branko$ php bin/console generate:bundle --namespace=Foggyline/SalesBundle

  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/SalesBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineSalesBundle.

Bundle name [FoggylineSalesBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

[Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/SalesBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```



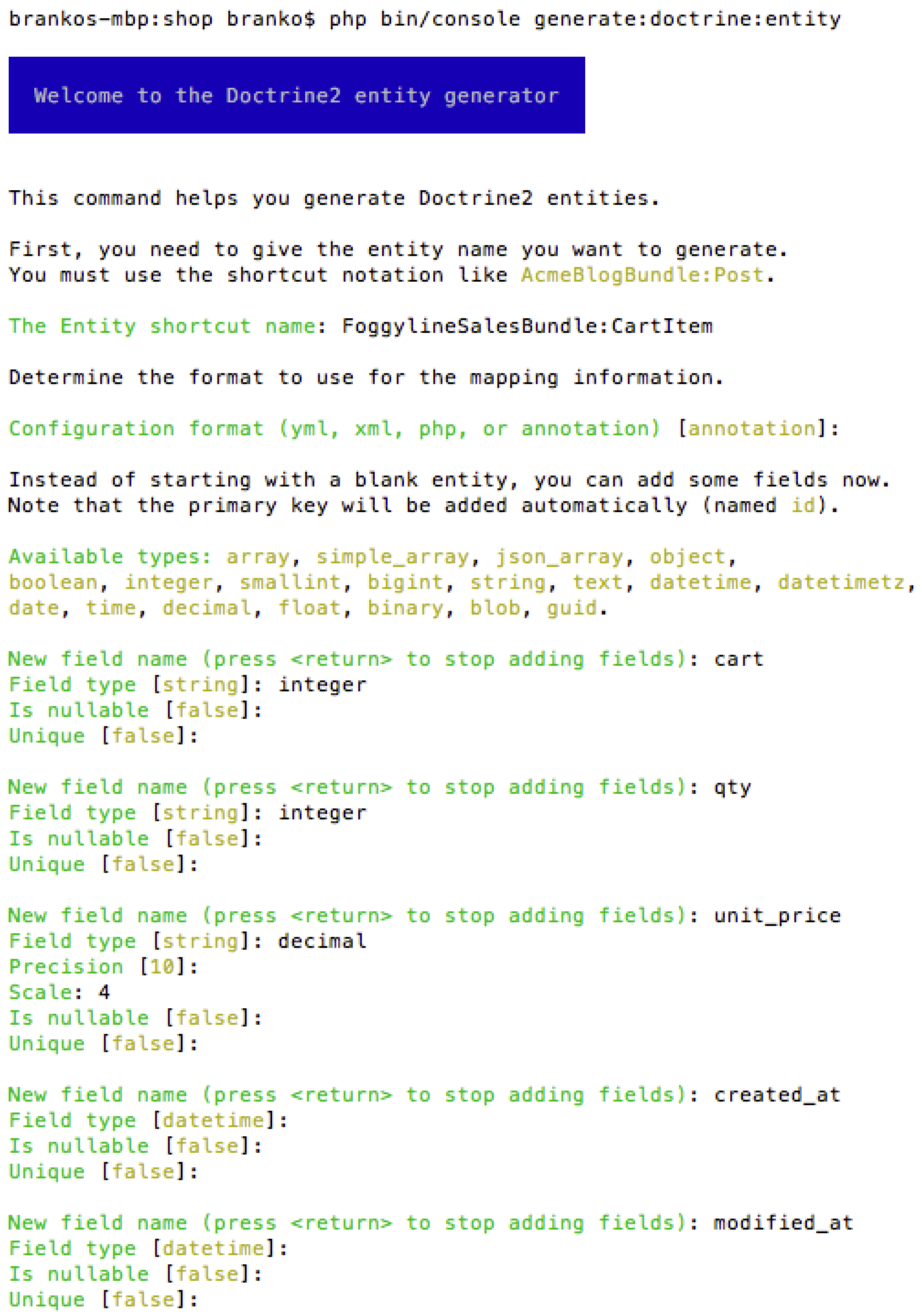
```
Brankos-MacBook-Pro:shop branko$ php bin/console generate:bundle --namespace=Foggyline/SalesBundle

  Welcome to the Symfony bundle generator!

Are you planning on sharing this bundle across multiple applications? [no]: yes

Your application code must be written in bundles. This command helps
you generate them easily.

Each bundle is hosted under a namespace (like Acme/BlogBundle).
The namespace should begin with a "vendor" name like your company name, your
project name, or your client name, followed by one or more optional category
sub-namespaces, and it should end with the bundle name itself
(which must have Bundle as a suffix).

See http://symfony.com/doc/current/cookbook/bundles/best_practices.html#bundle-name for more
details on bundle naming conventions.

Use / instead of \  for the namespace delimiter to avoid any problem.

Bundle namespace [Foggyline/SalesBundle]:

In your code, a bundle is often referenced by its name. It can be the
concatenation of all namespace parts but it's really up to you to come
up with a unique name (a good practice is to start with the vendor name).
Based on the namespace, we suggest FoggylineSalesBundle.

Bundle name [FoggylineSalesBundle]:

Bundles are usually generated into the src/ directory. Unless you're
doing something custom, hit enter to keep this default!

Target Directory [src/]:

What format do you want to use for your generated configuration?

Configuration format (annotation, yml, xml, php) [xml]:

  Bundle generation

> Generating a sample bundle skeleton into src/Foggyline/SalesBundle OK!
> Checking that the bundle is autoloaded: OK
> Enabling the bundle inside app/AppKernel.php: OK
> Importing the bundle's routes from the app/config/routing.yml file: OK

  Everything is OK! Now get to work :).
```



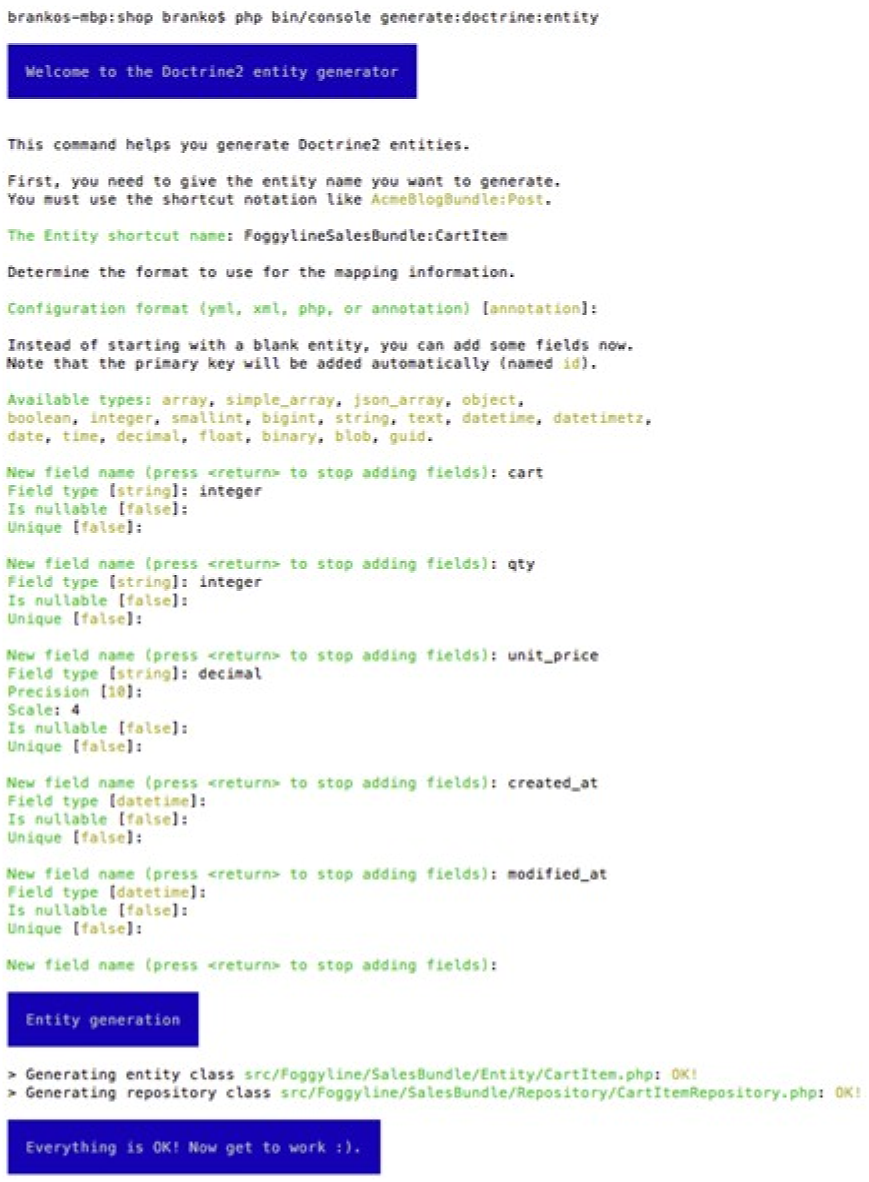
```
 You have new mail in /var/mail/branko
[Brankos-MacBook-Pro:shop branko$ php bin/console generate:doctrine:entity

   Welcome to the Doctrine2 entity generator


 This command helps you generate Doctrine2 entities.

 First, you need to give the entity name you want to generate.
 You must use the shortcut notation like AcmeBlogBundle:Post.

[The Entity shortcut name: FoggylineSalesBundle:Cart

 Determine the format to use for the mapping information.

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

 Instead of starting with a blank entity, you can add some fields now.
 Note that the primary key will be added automatically (named id).

 Available types: array, simple_array, json_array, object,
 boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
 date, time, decimal, float, binary, blob, guid.

 New field name (press <return> to stop adding fields): customer
[Field type [string]: integer
[Is nullable [false]:
[Unique [false]:

 New field name (press <return> to stop adding fields): created_at
[Field type [datetime]:
[Is nullable [false]:
[Unique [false]:

 New field name (press <return> to stop adding fields): modified_at
[Field type [datetime]:
[Is nullable [false]:
[Unique [false]:

 New field name (press <return> to stop adding fields):

   Entity generation

 > Generating entity class src/Foggyline/SalesBundle/Entity/Cart.php: OK!
 > Generating repository class src/Foggyline/SalesBundle/Repository/CartRepository.php: OK!
```



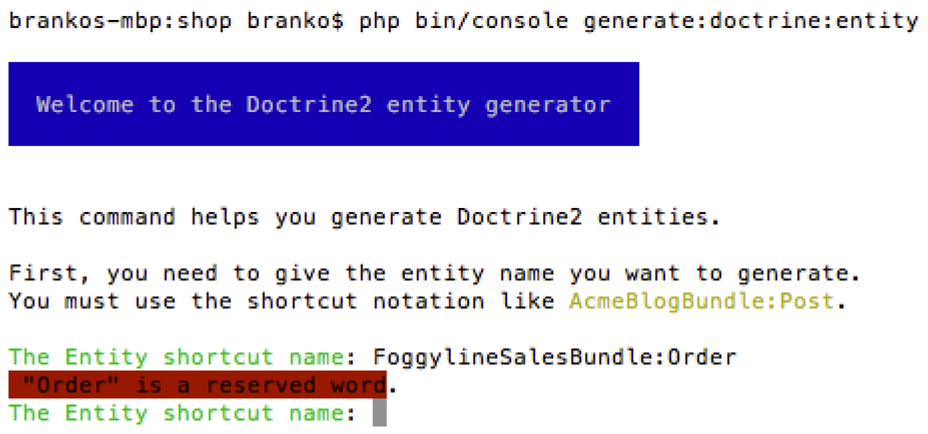
```
You have new mail in /var/mail/branko
Brankos-MacBook-Pro:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineSalesBundle:Cart

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): customer
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class src/Foggyline/SalesBundle/Entity/Cart.php: OK!
> Generating repository class src/Foggyline/SalesBundle/Repository/CartRepository.php: OK!

  Everything is OK! Now get to work :).
```



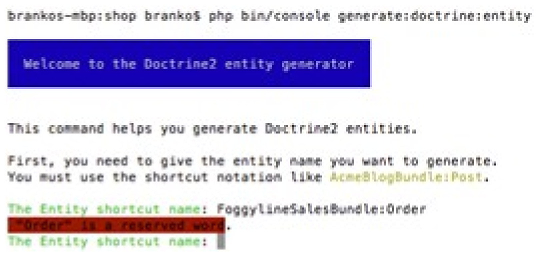
```
brankos-mbp:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineSalesBundle:CartItem

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): cart
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): qty
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): unit_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:
```



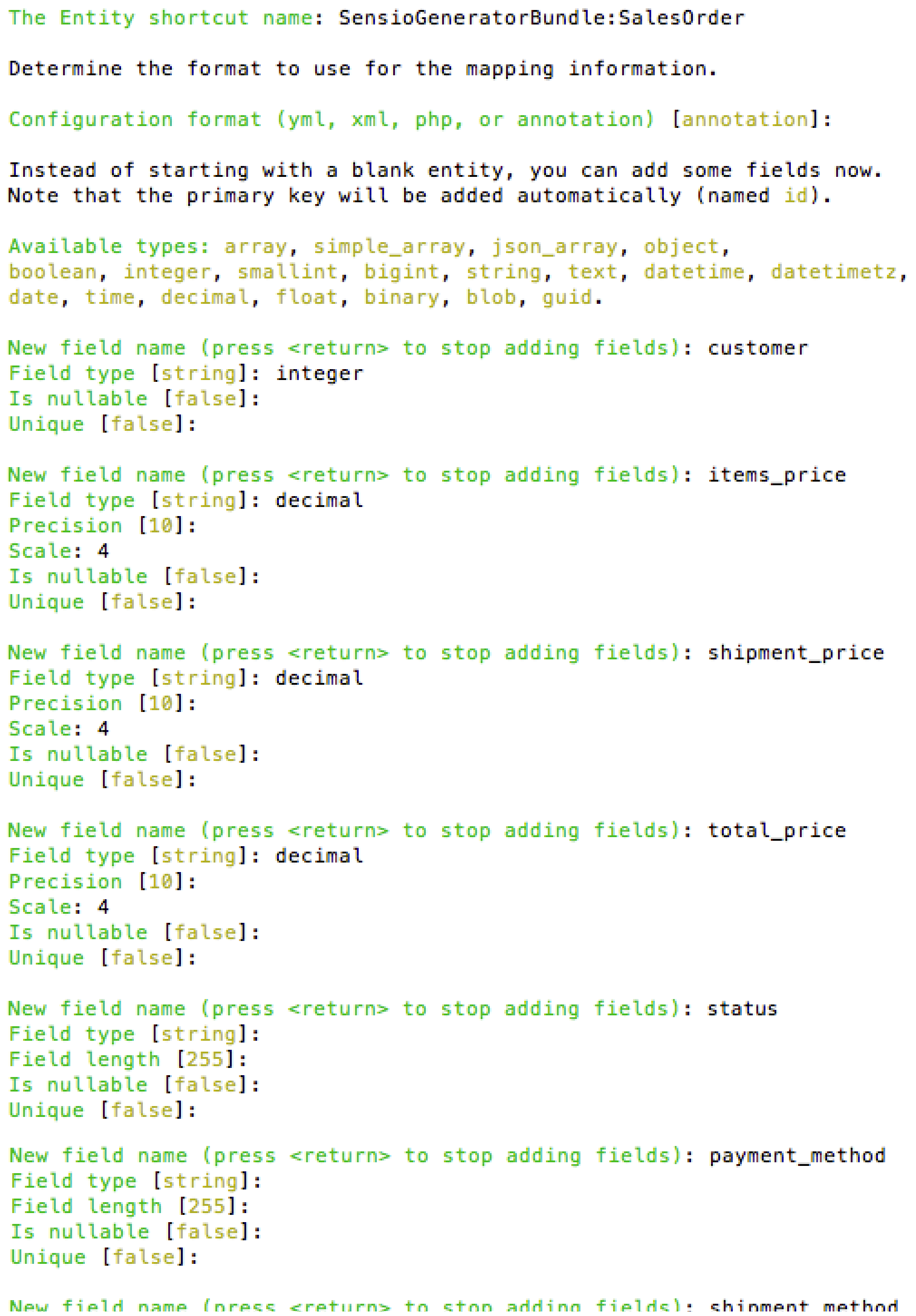
```
brankos-mbp:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator

This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineSalesBundle:CartItem

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): cart
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): qty
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): unit_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class src/Foggyline/SalesBundle/Entity/CartItem.php: OK!
> Generating repository class src/Foggyline/SalesBundle/Repository/CartItemRepository.php: OK!

  Everything is OK! Now get to work :).
```



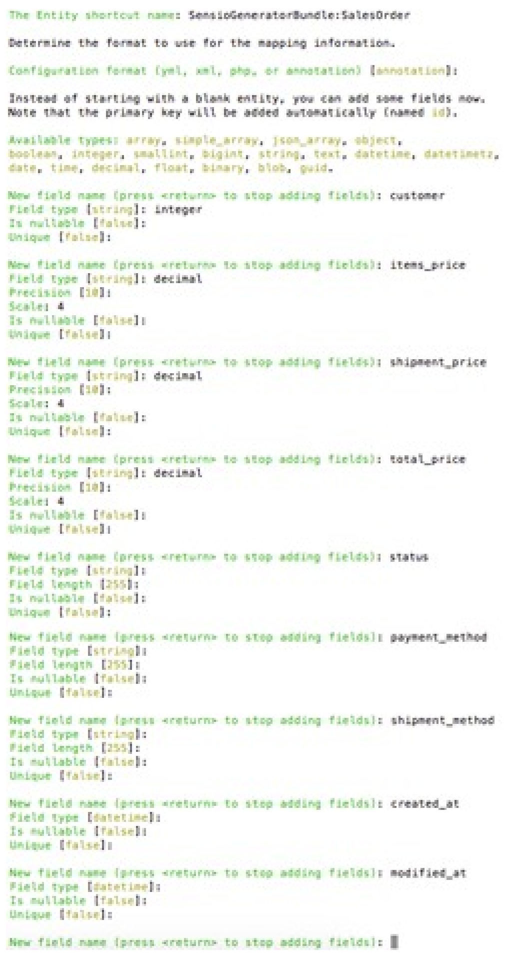
```
brankos-mbp:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineSalesBundle:Order
 "Order" is a reserved word.
The Entity shortcut name: 
```



```
brankos-mbp:shop branko$ php bin/console generate:doctrine:entity

  Welcome to the Doctrine2 entity generator


This command helps you generate Doctrine2 entities.

First, you need to give the entity name you want to generate.
You must use the shortcut notation like AcmeBlogBundle:Post.

The Entity shortcut name: FoggylineSalesBundle:Order
 "Order" is a reserved word.
The Entity shortcut name: 
```

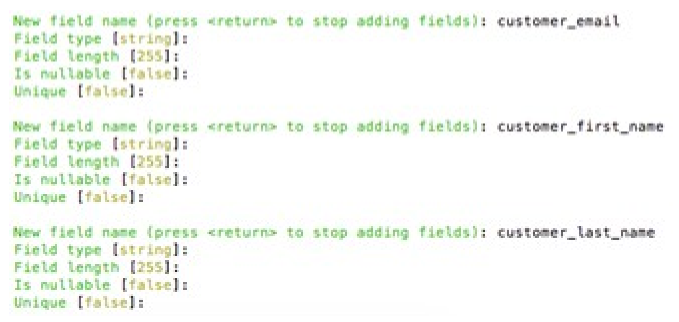


```
The Entity shortcut name: SensioGeneratorBundle:SalesOrder

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): customer
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): items_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): shipment_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): total_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): status
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): payment_method
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): shipment_method
```

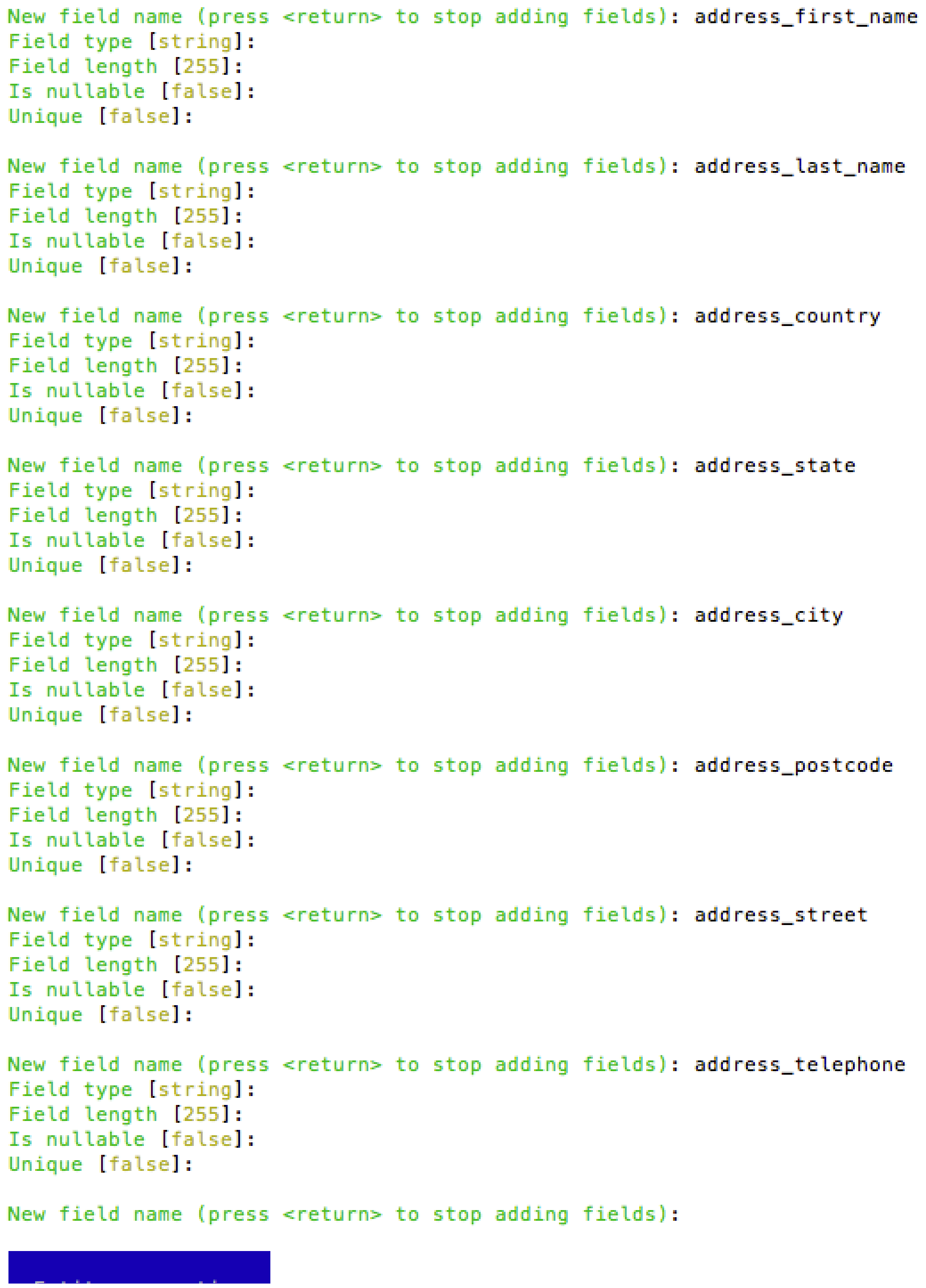


```
The Entity shortcut name: SensioGeneratorBundle:SalesOrder

Determine the format to use for the mapping information.

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

Instead of starting with a blank entity, you can add some fields now.
Note that the primary key will be added automatically (named id).

Available types: array, simple_array, json_array, object,
boolean, integer, smallint, bigint, string, text, datetime, datetimetz,
date, time, decimal, float, binary, blob, guid.

New field name (press <return> to stop adding fields): customer
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): items_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): shipment_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): total_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): status
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): payment_method
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): shipment_method
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):
```

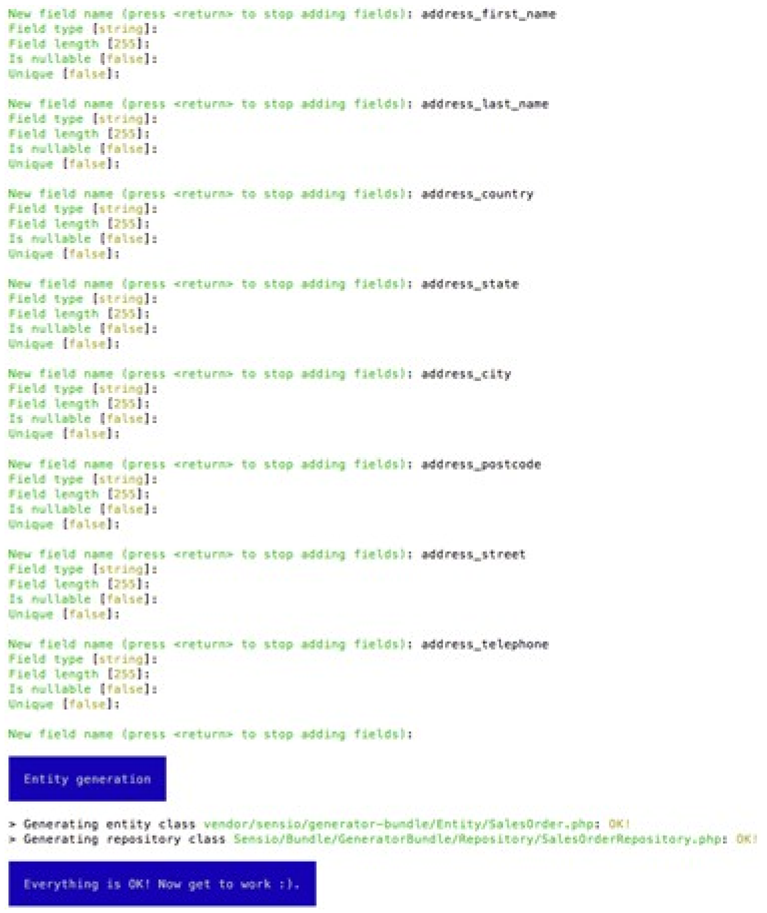
```
New field name (press <return> to stop adding fields): customer_email
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): customer_first_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): customer_last_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:
```



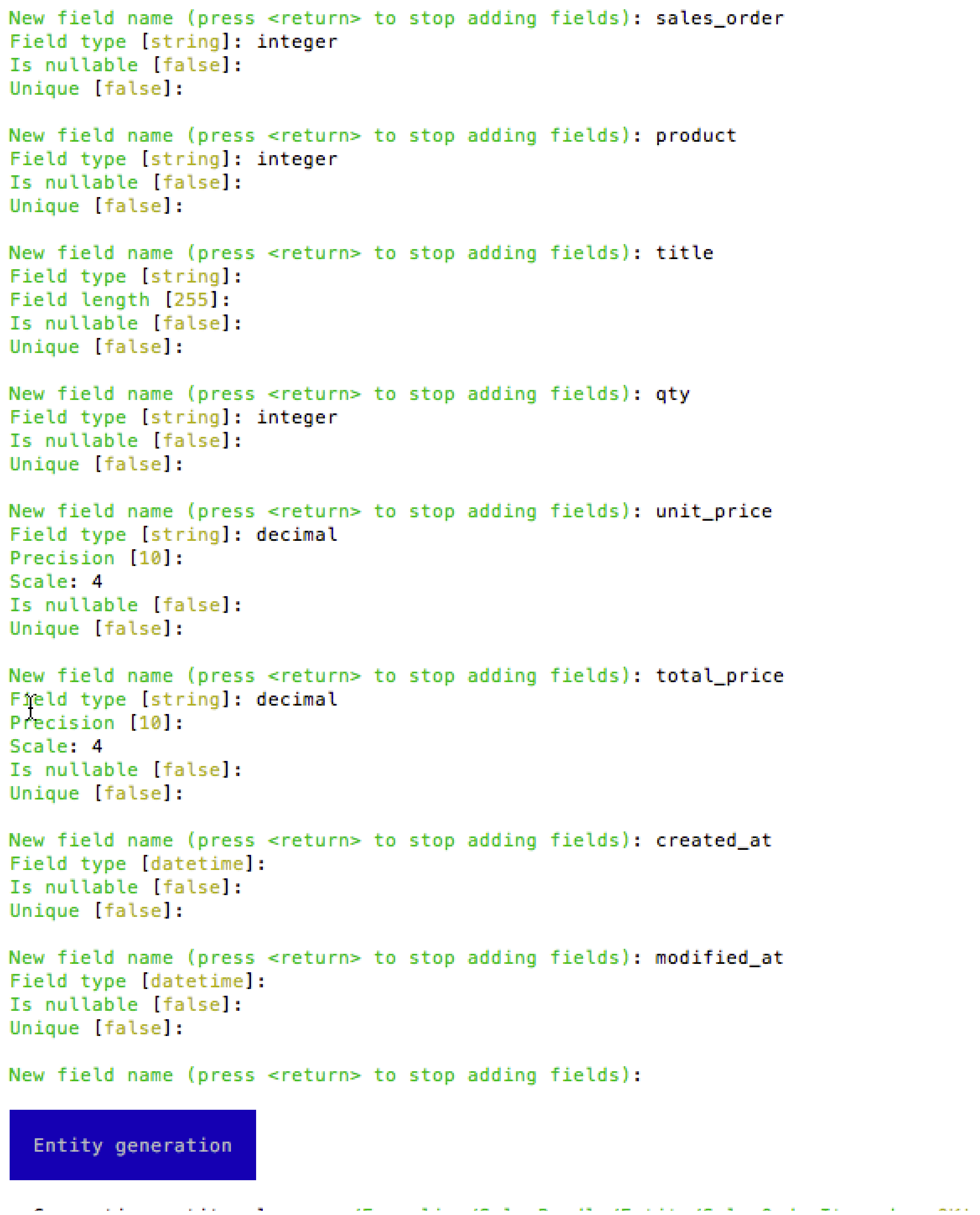
```
New field name (press <return> to stop adding fields): customer_email
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): customer_first_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): customer_last_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:
```



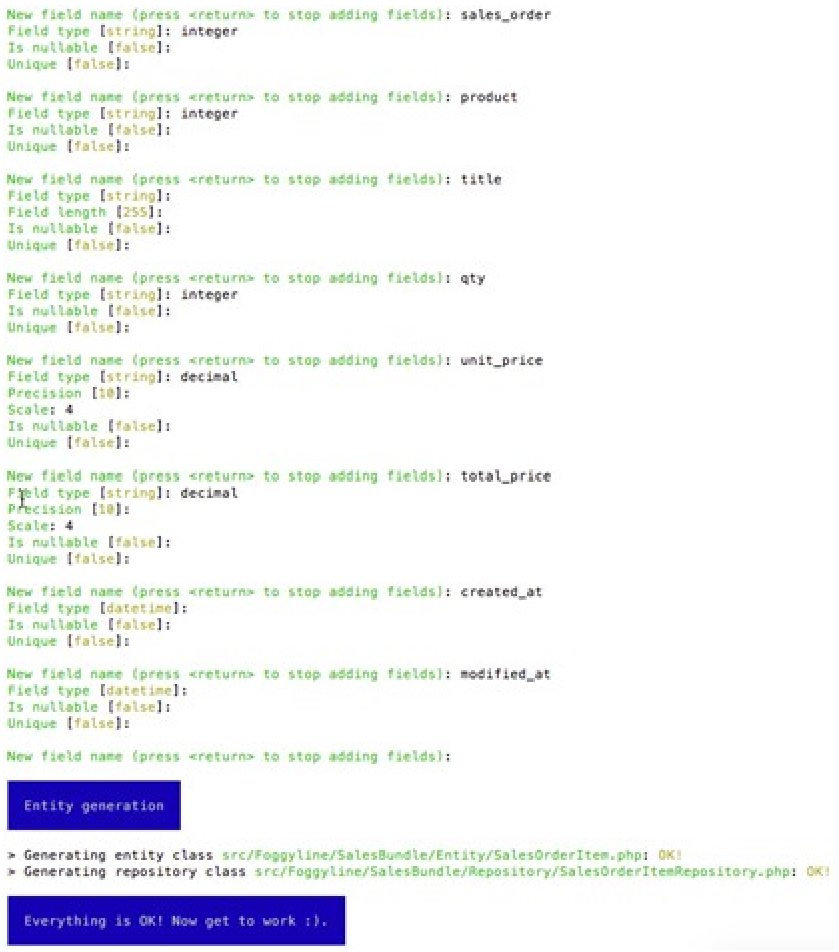
```
New field name (press <return> to stop adding fields): address_first_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_last_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_country
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_state
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_city
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_postcode
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_street
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_telephone
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):
```



```
New field name (press <return> to stop adding fields): address_first_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_last_name
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_country
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_state
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_city
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_postcode
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_street
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): address_telephone
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class vendor/sensio/generator-bundle/Entity/SalesOrder.php: OK!
> Generating repository class Sensio/Bundle/GeneratorBundle/Repository/SalesOrderRepository.php: OK!

  Everything is OK! Now get to work :).
```

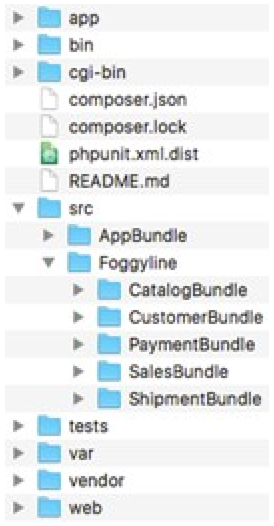


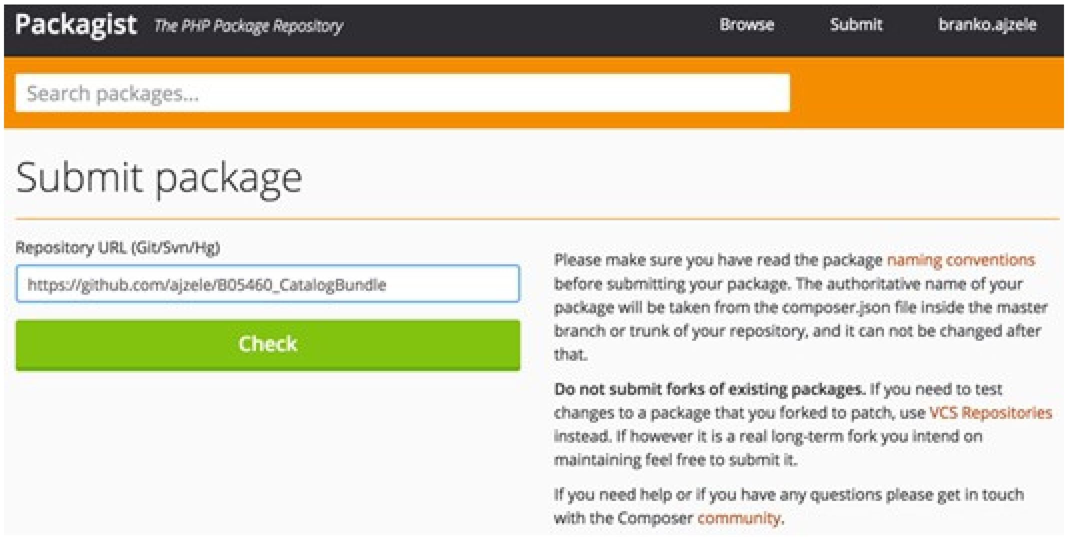
```
New field name (press <return> to stop adding fields): sales_order
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): product
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): title
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): qty
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): unit_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): total_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation
```



```
New field name (press <return> to stop adding fields): sales_order
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): product
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): title
Field type [string]:
Field length [255]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): qty
Field type [string]: integer
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): unit_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): total_price
Field type [string]: decimal
Precision [10]:
Scale: 4
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): created_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields): modified_at
Field type [datetime]:
Is nullable [false]:
Unique [false]:

New field name (press <return> to stop adding fields):

  Entity generation

> Generating entity class src/Foggyline/SalesBundle/Entity/SalesOrderItem.php: OK!
> Generating repository class src/Foggyline/SalesBundle/Repository/SalesOrderItemRepository.php: OK!

  Everything is OK! Now get to work :).
```

Chapter 12: Integrating and Distributing Modules





Submit package

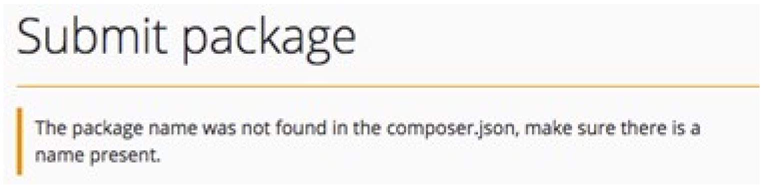
Repository URL (Git/Svn/Hg)

https://github.com/ajzele/B05460_CatalogBundle

Please make sure you have read the package naming conventions before submitting your package. The authoritative name of your package will be taken from the composer.json file inside the master branch or trunk of your repository, and it can not be changed after that.

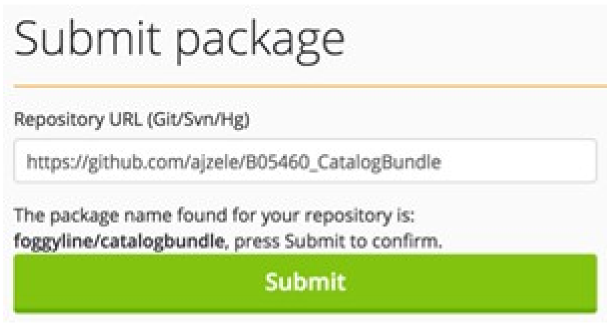
Do not submit forks of existing packages. If you need to test changes to a package that you forked to patch, use VCS Repositories instead. If however it is a real long-term fork you intend on maintaining feel free to submit it.

If you need help or if you have any questions please get in touch with the Composer community.



Submit package

The package name was not found in the composer.json, make sure there is a name present.

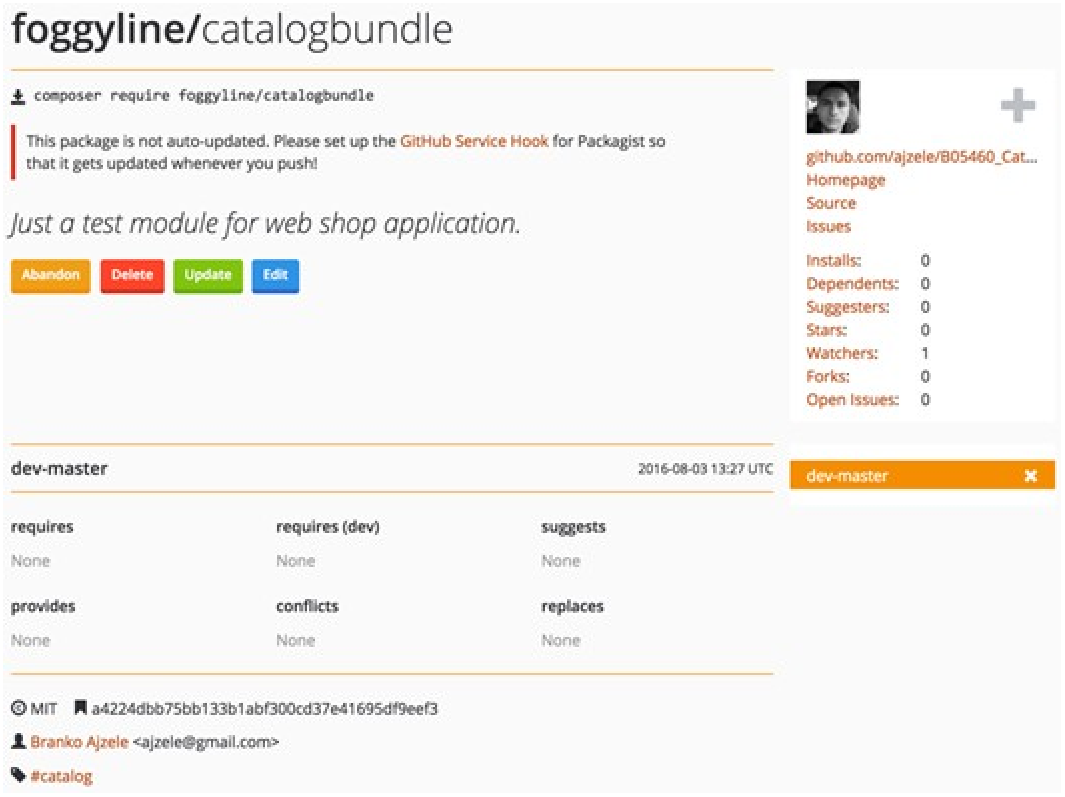


Submit package

Repository URL (Git/Svn/Hg)

https://github.com/ajzele/B05460_CatalogBundle

The package name found for your repository is: **foggyline/catalogbundle**, press Submit to confirm.



foggyline/catalogbundle

```
composer require foggyline/catalogbundle
```

This package is not auto-updated. Please set up the GitHub Service Hook for Packagist so that it gets updated whenever you push!

Just a test module for web shop application.

github.com/ajzele/B05460_Cat...
Homepage
Source
Issues

Installs:	0
Dependents:	0
Suggesters:	0
Stars:	0
Watchers:	1
Forks:	0
Open Issues:	0

dev-master 2016-08-03 13:27 UTC

requires	requires (dev)	suggests
None	None	None
provides	**conflicts**	**replaces**
None	None	None

MIT a4224dbb75bb133b1abf300cd37e41695df9eef3

Branko Ajzele <ajzele@gmail.com>

#catalog