Fecshop 介绍

源文档：<http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_product_image.html>

Fecshop，是由Terry从2015年初开始，一直坚持到今天的开源电商项目，非商业化运作，是真正 开源的电商系统， 遵循BSD-3-Clause开源协议，您可以免费的应用于您的线上商城。如有任何其他疑问， 可以联系邮箱：2358269014@qq.com

Fecshop 全称为Fancy ECommerce Shop，是基于php Yii2框架之上开发的一款优秀的开源电商系统， 遵循BSD-3-Clause开源协议，Fecshop支持多语言，多货币，架构上支持pc，手机web，手机app，和erp对接等入口，您可以免费快速的定制和部署属于您的电商系统。 截止到2017-12-15，appfront（pc前端web），appadmin（后台），apphtml5（wap端web）， appserver（VUE，手机app的api提供端），console（命令行）， appapi（和第三方系统通信端）都已经完成，现在已经趋于稳定， Terry在2010年开始进入跨境电商领域， 用了不少开源电商系统，譬如magento， 发现开源框架都有一定 的诟病，在并发方面差，后期扩展，业务发展后期重构难， 尤其是现在的移动端的发展，多入口的电商模式占据主流， 性能方面的要求越来越高，Fecshop基于Yii2的高效框架，在此基础上进一步封装，加入了 service层和block层，数据库采用了nosql和mysql结合的方式， 关系型表放到mysql中，譬如优惠券，购物车，订单等， 非关系型数据表（非关系型代表不会出现多表强事务类型操作） 放到mongodb中， 缓存用redis，搜索目前用的是mongodb的FullTextSearch功能， 支持一些主流语言的分词与搜索，对于中文搜索，使用的是xunsearch， 后期会扩展ElasticSearch来进行搜索（ElasticSearch有中文插件，安装后支持中文分词）。 总之，Fecshop目前的定位是为了让程序员们有一个方便学习，扩展，开发的电商框架系统， 如果您发现有哪些代码结构可以优化，调整，或者您有更加合理的建议，可以发送到邮箱： 2358269014@qq.com。

**Fecshop线上案例**：<http://www.fecshop.com/topic/55>

**关于Fecshop**：<http://www.fecshop.com/first>

**Fecshop的各种问题解答QA**：<http://www.fecshop.com/topic/400>

**Fecshop 【Appfront】PC Web Demo**：<http://fecshop.appfront.fancyecommerce.com/>

**Fecshop 【Apphtml5】Mobile Web Demo（WAP）**：[http://fecshop.apphtml5.fancyecommerce.com](http://fecshop.apphtml5.fancyecommerce.com/)

**Fecshop 【Appserver】Mobile VUE Demo(Appserver,前后端彻底分离模式)**：<http://demo.fancyecommerce.com/#/>

**Fecshop 开源协议**：<http://www.fecshop.com/license>

**Fecshop 使用说明文档地址：** <http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-README.html>

**Fecshop 二次开发文档地址：** <http://www.fecshop.com/doc/fecshop-guide/develop/cn-1.0/guide-README.html>

**FecShop Github地址：**<https://github.com/fancyecommerce/yii2_fecshop>

**FecShop oschina地址：** <https://www.oschina.net/p/fecshop>

**Terry个人博客：**<http://www.fancyecommerce.com/>

**TerryCSDN博客：**<http://blog.csdn.net/terry_water>

# Fecshop 配置结构

1. [配置文件以及优先级](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_config_construction.html)
2. [优先级部分配置细化 - yii\helpers\ArrayHelper::merge()部分](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_config_construction.html#yiihelpersarrayhelpermerge)
3. [配置加速](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_config_construction.html)

fecshop 配置结构是层级覆盖的方式，通过yii2的 yii\helpers\ArrayHelper::merge()函数 将所有的配置进行合并，最终成为一个大数组。配置结构通过层级覆盖的方式可以很方便的对fecshop的功能进行重写

## 配置文件以及优先级

#### 1.fecshop合并的配置文件：

fecshop配置是根据优先级，优先级高的配置可以覆盖优先级低的配置， 优先级由低到高的排列如下（其实就是index.php中的代码）：

$config = yii\helpers\ArrayHelper::merge(

require(\_\_DIR\_\_ . '/../../common/config/main.php'),

require(\_\_DIR\_\_ . '/../../common/config/main-local.php'),

require(\_\_DIR\_\_ . '/../config/main.php'),

require(\_\_DIR\_\_ . '/../config/main-local.php'),

# fecshop services config

require(\_\_DIR\_\_ . '/../../vendor/fancyecommerce/fecshop/config/fecshop.php'),

# fecshop module config

require(\_\_DIR\_\_ . '/../../vendor/fancyecommerce/fecshop/app/appfront/config/appfront.php'),

# thrid part confing

# common modules and services.

require(\_\_DIR\_\_ . '/../../common/config/fecshop\_local.php'),

# appadmin local modules and services.

require(\_\_DIR\_\_ . '/../config/fecshop\_local.php')

);

1到4，是Yii2框架的配置文件结构：

1.@common/config/main.php # Yii2 公用配置，一般是Yii2的一些组件的公用配置。

2.@common/config/main-local.php # Yii2 本地公用配置，一般是Yii2的一些数据库组件的配置

3.@app/config/main.php # Yii2 本app内的配置

4.@app/config/main-local.php # Yii2 本app内的本地配置

5和6部分，是fecshop的配置文件结构：

5.@vendor/fancyecommerce/fecshop/config/fecshop.php # Fecshop的公用配置，一般是组件和服务的配置。

6.@vendor/fancyecommerce/fecshop/app/appfront/config/appfront.php # Fecshop appfront这个入口的独有配置，里面包含模块和一些组件的配置。

第7部分为第三方开发的扩展的配置引入部分，您使用的第三方插件的配置，可以在这里引入进来。

7.第三方配置文件地址。

8和9部分，是二次开发者的配置文件部分：

8.@common/config/fecshop\_local.php # 二次开发者的各个入口的公用配置

9.@app/config/fecshop\_local.php # 二次开发者的当前入口的独有配置

上面8个配置文件的内容，通过yii\helpers\ArrayHelper::merge()函数合并，最后 形成一个数组，在合并的过程中，1的优先级最低，8个优先级最高，优先级高的配置 项可以覆盖优先级低的配置项。

覆盖顺序：

1.fecshop（5-6）可以覆盖前面Yii2的配置项(1-4)

2.第三方扩展（7），可以覆盖fecshop的配置项(1-6)

3.二次开发者（8-9）可以覆盖上面所有的配置(1-7)

#### 2.对于运费的配置，是通过csv的方式存储公式，路径为

@common/config/shipping/fast\_shipping.csv

#### 3.yii class map重写机制

$yiiClassMap = require(\_\_DIR\_\_ . '/../config/YiiClassMap.php');

if(is\_array($yiiClassMap) && !empty($yiiClassMap)){

foreach($yiiClassMap as $namespace => $filePath){

Yii::$classMap[$namespace] = $filePath;

}

}

通过classMap的配置，可以重写任意一个class文件，譬如

<?php

return [

'fecshop\app\appfront\helper\test\My' => '@appfront/helper/My.php',

];

具体使用方法参看文章：[通过配置的方式重写某个Yii2 文件 或第三方扩展文件](http://www.fancyecommerce.com/2016/10/13/%e9%80%9a%e8%bf%87%e9%85%8d%e7%bd%ae%e7%9a%84%e6%96%b9%e5%bc%8f%e9%87%8d%e5%86%99%e6%9f%90%e4%b8%aayii2-%e6%96%87%e4%bb%b6-%e6%88%96%e7%ac%ac%e4%b8%89%e6%96%b9%e6%89%a9%e5%b1%95%e6%96%87%e4%bb%b6/)

## 优先级部分配置细化 - yii\helpers\ArrayHelper::merge()部分

上面的9个配置文件，有的配置文件还包含其他的配置文件，下面逐一列举

### 1.@vendor/fancyecommerce/fecshop/config/fecshop.php

该配置文件包含:

1.1@vendor/fancyecommerce/fecshop/config/services/\*.php ,这个文件夹下面所有的php配置文件，这里面是各个Fecshop Services（服务）的配置

1.2@vendor/fancyecommerce/fecshop/config/components/\*.php ，这个文件夹下面的所有的php文件，这里面是各个Fecshop Component（组件）的配置

### 2.@vendor/fancyecommerce/fecshop/app/appfront/config/appfront.php

该配置文件包含：

2.1@vendor/fancyecommerce/fecshop/app/appfront/config/params.php 这里面是一些参数的默认配置。

2.2@vendor/fancyecommerce/fecshop/app/appfront/config/modules/\*.php 这个文件夹下面所有的php配置文件，这里面是各个模块对应的配置

### 3.@common/config/fecshop\_local.php

该配置文件包含：

3.1@common/config/fecshop\_local\_modules/\*.php 这个文件夹下面所有的php配置文件，这里面是二次开发的公用部分，添加或重写的 模块方面的配置

3.2@common/config/fecshop\_local\_services/\*.php 这个文件夹下面所有的php配置文件，这里面是二次开发的公用部分，添加或重写的 服务方面的配置

4.@app/config/fecshop\_local.php

该配置文件包含：

4.1@app/config/fecshop\_local\_modules/\*.php 这个文件夹下面所有的php配置文件，这里面是二次开发的公用部分，添加或重写的 模块方面的配置

4.2@app/config/fecshop\_local\_services/\*.php 这个文件夹下面所有的php配置文件，这里面是二次开发的公用部分，添加或重写的 服务方面的配置

一开始说的9个配置文件，以及对应的这些细分的配置文件，共同组合起来合并，形成 fecshop的最终配置。

通过配置项的覆盖，我们就可以重写各个组件，服务等功能，二次开发自己的功能。

## 配置加速

配置文件的合并是在index.php文件中， 为了加速，您可以把合并后的数组放到缓存中，然后直接读取出来，节省数据合并带来的开销。 如果您更新了配置，您需要刷新一下缓存让更新的配置生效。

# Fecshop 手机检测跳转

当手机访问pc端(appfront)url，可以通过设置自动跳转到手机html5（apphtml5）端。

### 配置设置

打开配置文件 @appfront/config/fecshop\_local\_services/Store.php

可以看到配置代码：

'fecshop.appfront.fancyecommerce.com' => [

'language' => 'en\_US', // 语言简码需要在@common/config/fecshop\_local\_services/FecshopLang.php 中定义。

'languageName' => 'English', // 语言简码对应的文字名称，将会出现在语言切换列表中显示。

'localThemeDir' => '@appfront/theme/terry/theme01', // 设置当前store对应的模板路径。关于多模板的方面的知识，您可以参看fecshop多模板的知识。

'thirdThemeDir' => [], // 第三方模板路径，数组，可以多个路径

'currency' => 'USD', // 当前store的默认货币,这个货币简码，必须在货币配置中配置

/\*

\* 当设备满足什么条件的时候，进行跳转。

\* 这种方式不怎么高效，最好的方式是在nginx中配置。

\*/

'mobile' => [

'enable' => true,

'condition' => ['phone', 'tablet'], // phone 代表手机，tablet代表平板，当都填写，代表手机和平板都会进行跳转

'redirectDomain' => 'fecshop.apphtml5.fancyecommerce.com', // 如果是移动设备访问进行域名跳转，这里填写的值为store key

'https' => false, // 手机端url是否支持https,如果支持，设置https为true，如果不支持，设置为false

],

当 mobile.enable 设置为true，当访问pc端会跳转到html5端。

### 原理

打开@fecshop/services/Store.php ，您可以看到 html5DevideCheckAndRedirect()方法：

/\*\*

\* @property $store\_code | String

\* @property $store | Array

\* mobile devide url redirect.

\*/

protected function html5DevideCheckAndRedirect($store\_code, $store)

{

if (!isset($store['mobile'])) { //检查有没有配置html5跳转参数

return;

}

$mobileDetect = Yii::$service->helper->mobileDetect;

$enable = isset($store['mobile']['enable']) ? $store['mobile']['enable'] : false;

if (!$enable) { //检查有没有启用html5跳转

return;

}

$condition = isset($store['mobile']['condition']) ? $store['mobile']['condition'] : false; //获取跳转条件，如果没有设置跳转条件，默认为false

$redirectDomain = isset($store['mobile']['redirectDomain']) ? $store['mobile']['redirectDomain'] : false; //获取跳转地址,默认为false

if (is\_array($condition) && !empty($condition) && !empty($redirectDomain)) { //当满足跳转条件存在，并且跳转地址存在时，进行以下步骤...

$mobile\_https = (isset($store['mobile']['https']) && $store['mobile']['https']) ? true : false;

if (in\_array('phone', $condition) && in\_array('tablet', $condition)) {

if ($mobileDetect->isMobile()) { //检查是不是手机

$this->redirectMobile($store\_code, $redirectDomain, $mobile\_https);

}

} elseif (in\_array('phone', $condition)) {

if ($mobileDetect->isMobile() && !$mobileDetect->isTablet()) { //检查是不是平板电脑

$this->redirectMobile($store\_code, $redirectDomain, $mobile\_https);

}

} elseif (in\_array('tablet', $condition)) {

if ($mobileDetect->isTablet()) {

$this->redirectMobile($store\_code, $redirectDomain, $mobile\_https);

}

}

}

}

/\*\*

\* @property $store\_code | String

\* @property $redirectDomain | String

\* 设备满足什么条件的时候进行跳转。

\*/

protected function redirectMobile($store\_code, $redirectDomain, $mobile\_https)

{

$currentUrl = Yii::$service->url->getCurrentUrl();

$redirectUrl = str\_replace($store\_code, $redirectDomain, $currentUrl);

// pc端跳转到html5，可能一个是https，一个是http，因此需要下面的代码进行转换。

if ($mobile\_https) {

if (strstr($redirectUrl,'https://') || strstr($redirectUrl,'http://')) {

$redirectUrl = str\_replace('http://','https://',$redirectUrl);

} else {

$redirectUrl = 'https:'.$redirectUrl;

}

} else {

if (strstr($redirectUrl,'https://') || strstr($redirectUrl,'http://')) {

$redirectUrl = str\_replace('https://','http://',$redirectUrl);

} else {

$redirectUrl = 'http:'.$redirectUrl;

}

}

header('Location:'.$redirectUrl);

}

html5DevideCheckAndRedirect方法检测是否是手机访问，如果是，则进行跳转。

# Fecshop 货币

1. [货币配置](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_currency.html)

fecsop 支持多货币

## 货币配置

您可以在@common/config/fecshop\_local\_services/Page.php中进行配置货币。

'currency' => [

'baseCurrecy' => 'USD', # 基础货币，后台产品的价格都使用基础货币填写价格值。

'defaultCurrency' => 'USD', # 默认货币，如果store不设置货币，就使用这个store默认货币

'currencys' => [

'USD' => [ # 货币简码，USD代表美元，这个是国际标准

'rate' => 1, # 汇率 当前货币/基础货币的比值，譬如，人民币/美元 = 7

'symbol' => '$', #货币符号

],

'EUR' => [ # 欧元

'rate' => 0.93,# 汇率

'symbol' => '€',

],

//'AUD' => [

// 'rate' => 1.33,

// 'symbol' => 'AU$',

//],

'GBP' => [ # 英镑

'rate' => 0.8,

'symbol' => '£',

],

'CNY' => [ # 人民币

'rate' => 6.87,

'symbol' => '￥',

],

],

],

设置货币后，在前端的顶部的语言切换选项中，就能看到您添加或者修改的货币配置。

另外在store中设置的货币，必须在此处存在，否则将会报错。

# ﻿Fecshop Store

fecshop是支持多store的，store一般是用来构建多语言站点， 可以为每一个store指定不同的语言和url结构，设置不同的语言，默认货币,模板等。

### 1.Store的配置：

您可以在@appfront/config/fecshop\_local\_services/Store.php中配置Store

示例代码：

'stores' => [

# store key：域名去掉http部分，作为key，这个必须这样定义。

'fecshop.appfront.fancyecommerce.com' => [

'language' => 'en\_US', # 语言简码需要在@common/config/fecshop\_local\_services/FecshopLang.php 中定义。

'languageName' => 'English', # 语言简码对应的文字名称，将会出现在语言切换列表中显示。

'localThemeDir' => '@appfront/theme/terry/theme01', # 设置当前store对应的模板路径。关于多模板的方面的知识，您可以参看fecshop多模板的知识。

'thirdThemeDir' => [], # 第三方模板路径，数组，可以多个路径

'currency' => 'USD', # 当前store的默认货币,这个货币简码，必须在货币配置中配置

'mobile' => [ # 当设备满足什么条件的时候，进行跳转。

'enable' => true,

'condition' => ['phone','tablet'], # phone 代表手机，tablet代表平板

'redirectUrl' => 'fecshop.apphtml5.fancyecommerce.com', # 如果是移动设备访问进行域名跳转，这里填写的值为store key

],

# 第三方账号登录配置

'thirdLogin' => [

# facebook账号登录

'facebook' =>[ #fb api配置 ，fb可以一个app设置pc和手机web两个域名

'facebook\_app\_id' => '1849609081926823',

'facebook\_app\_secret' => '2e097a6d5a424531770fc05760dd7139',

],

# google账号登录

"google" => [ #谷歌api visit https://code.google.com/apis/console to generate your google api

'CLIENT\_ID' => '380372364773-qdj1seag9bh2n0pgrhcv2r5uoc58ltp3.apps.googleusercontent.com',

'CLIENT\_SECRET' => 'ei8RaoCDoAlIeh1nHYm0rrwO',

],

]

],

需要注意的是您的store key，如果您的首页是 http://fecshop.appfront.fancyecommerce.com，那么您的key为fecshop.appfront.fancyecommerce.com ，如果您的首页为 http://fecshop.appfront.fancyecommerce.com/fr，那么您的key为 fecshop.appfront.fancyecommerce.com/fr

您可以为您的每一个store指定特定的域名，用不同的子域名或用不同的语言路径的 方式，来构建您的多语言的网站。

### 2.多语言站点的构建方式：

#### 1.子域名的方式构建多语言站点，譬如：

www.fecshop.com（英文）,fr.fecshop.com（法文） de.fecshop.com（德文）

这种方式可以在@app/web/index.php中设置 ini\_set('session.cookie\_domain', '.fancyecommerce.com'); 来进行子域名之间session的共享。

#### 2.url路径的方式构建多语言站点，譬如：

www.fecshop.com（英文）,www.fecshop.com/fr/（法文）,www.fecshop.com/de/（德文） 这种多语言方式因为都是同样的域名，只是后缀不同，因此， 不需要像第1步那样做session共享

#### 3.独立多域名方式:

www.fecshop.com（英文）,www.fecshop.fr（法文）,www.fancyecommerce.de（德文）

这种方式，session不能共享（当然，您可以做跨域session共享，但是是比较麻烦的， 因此，建议使用第一种，子域的方式），因此购物车和登录信息在切换站点的时候，需要重新 进行登录。

### 4.store原理

4.1 在入口文件 @app/web/index.php ，可以加载所有的配置文件，合并

4.2 对于appfront appserver apphtml5这三个入口是需要store的， 因此需要在Yii2 bootstrap初始化过程中就开始执行，下面以appfront入口为例子讲解

@fecshop/app/appfront/config/appfront.php ， 可以看到配置：

'bootstrap' => ['store'],

也就是在初始化的时候执行store component，关于Yii2的bootstrap可以查看资料： [启动引导（Bootstrapping）](http://www.yiichina.com/doc/guide/2.0/runtime-bootstrapping) , [yii2 初始化的bootstrap过程 -引导](http://www.fancyecommerce.com/2016/05/18/yii2-%E5%88%9D%E5%A7%8B%E5%8C%96%E7%9A%84bootstrap%E8%BF%87%E7%A8%8B-%E5%BC%95%E5%AF%BC/)

下面我们去找store component

4.3 打开配置文件： @fecshop/config/components/Store.php 文件可以看到：

return [

'store' => [

'class' => 'fecshop\components\Store',

],

];

通过配置文件，我们找到store component , @fecshop\components\Store.php, 代码如下：

class Store extends Component implements BootstrapInterface

{

public function bootstrap($app)

{

Yii::$service->store->bootstrap($app);

}

}

通过上面的代码，我们可以看到，就是执行store services 的bootstrap方法 ，通过配置文件：@fecshop/config/services/Store.php，我们可以看到

'store' => [

'class' => 'fecshop\services\Store',

我们打开 @fecshop\services\Store.php ,查看 bootstrap()方法， 可以看到很多store的初始化代码，譬如语言，货币等， 都是在这个函数里面进行的，您可以细看里面的代码，具体您可以查看这个文件，这里不做代码粘贴了。

# Fecshop 多语言

1. [一：语言的配置](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_mutil_lang.html)
2. [二：翻译](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_mutil_lang.html)
3. [三：各个入口的语言配置](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_mutil_lang.html)
4. [四：如何添加一门新语言](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/guide-fecshop_mutil_lang.html)

fecshop 支持多语言翻译，可以通过多个途径进行语言的切换，一共分为四个大部分： 数据库数据的翻译，网页内容数据的翻译，网站提示和报错等信息的翻译，邮件内容的 语言翻译，当您为一个store指定了语言，那么就会按照上面四种途径进行相应的语言翻译， ,进而形成对应语言的翻译语言。

## 一：语言的配置

在fecshop FecshopLang 服务 中可以配置语言选项，你可以在 @common/config/fecshop\_local\_services/FecshopLang.php中进行配置语言项，譬如：

'fecshoplang' => [

//'class' => 'fecshop\services\FecshopLang',

// mongoSearchLangName 在各个语言下字段参考资料如下：（不支持中文）

// https://docs.mongodb.com/manual/reference/text-search-languages/#text-search-languages

'allLangCode' => [

# 'en\_US' 是标准语言简码 code对应的值en取 “标准语言简码”的前两位字符，

# 该值设置后，进行了产品分类数据的添加后，不能修改，否则会出现部分翻译语言丢失。

'en\_US' => [

'code' => 'en',

],

'fr\_FR' => [

'code' => 'fr',

],

'de\_DE' => [

'code' => 'de',

],

'es\_ES' => [

'code' => 'es',

],

'ru\_RU' => [

'code' => 'ru',

],

'pt\_PT' => [

'code' => 'pt',

],

'zh\_CN' => [

'code' => 'zh',

],

],

# 默认语言。

'defaultLangCode' => 'en',

],

上面是fecshop默认的几个语言配置，您可以根据自己的业务需求进行语言的设置和添加。

fecshop的每个store设置的不同语言，以及对应的产品搜索(mongodb full search需要根据语言进行切词)，都要依赖上面的配置。

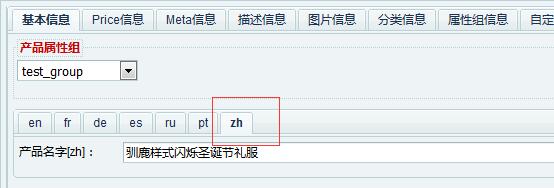
[语言简码表](http://blog.csdn.net/wed110/article/details/50886195)

## 二：翻译

### 数据库数据

数据库中的数据，譬如产品表中的产品名字，产品描述等，分类表中的分类名字和分类描述， 在数据库表中都是保存了多份语言数据





在网站前台展示数据的时候，根据当前的store设置的语言，将相应语言的数据取出来展示即可。 如果相应语言的值在数据库中没有添加，那么将会把默认语言的值取出来作为当前语言的值。

### 网页内容语言的翻译

网站内容语言的翻译，是依靠的翻译文件，在实现方面依赖的Yii::t()函数，关于 Yii2多语言翻译的知识，可以参看地址 [Yii2多语言](http://www.yiichina.com/doc/guide/2.0/tutorial-i18n)

### Fecshop的翻译文件路径为：（举例appfront入口）

@vendor/fancyecommerce/fecshop/app/appfront/languages/，在这个文件夹下面可以看到 各个语言的文件包，进入相应语言，就可以看到翻译文件，譬如中文的翻译文件路径为 ：@vendor/fancyecommerce/fecshop/app/appfront/languages/zh\_CN/appfront.php ，在这个文件里面就可以看到所有的中文翻译内容，如果您想重写或者添加新 的翻译，可以到 @appfront/languages/zh\_CN/appfront.php 中添加或者重写翻译 数组，注意，这里使用的是php的数组，需要按照相应格式填写，否则会报错。

在翻译文件中，您可能看到这样的带有{}的部分，譬如下面的 {passwdMinLength}, 这是一个动态变量，这个请不要翻译和做其他的任何改动，直接复制上去即可，譬如下面的 翻译，这个值是由php动态计算而来。

'Password length must be greater than or equal to {passwdMinLength}'

=> '密码长度必须大于或等于{passwdMinLength}',

'Password length must be less than or equal to {passwdMaxLength}'

=> '密码长度必须小于或等于{passwdMaxLength}',

'The passwords are inconsistent'

=> '密码不一致',

### 提示报错信息的翻译

在网站顶部，会出现一些提示信息和报错信息，譬如您注册邮箱的时候，会提示你的格式不正确等， 这些数据也是用翻译文件的方式，方法与上面类似，都是在在同一个文件中 添加翻译内容,方法和方式同上。

### 邮件内容的翻译

邮件内容的模板，是在theme中，譬如appfront的邮件模板，是在 @fecshop/app/appfront/theme/base/default/mailer下面,其中, @fecshop/app/appfront/theme是模板文件夹路径，base是模板包名， default是模板名，mailer是邮件theme文件夹，在mailer路径下，打开 customer/account/forgotpassword,里面是用户忘记密码发送的邮件， 其中subject\_开头的都是邮件的标题，body\_开头的都是邮件的内容 en结尾的代表英文语言的邮件，zh结尾的代表中文语言的邮件，打开相应文件 从en复制一份到对应的语言，然后修改里面的语言即可， 有一些语言，里面没有对应的文件，您新建即可。

添加或者修改邮件内容，举例：

如果您想修改 @fecshop/app/appfront/theme/base/default/mailer/customer/account/forgotpassword/body\_zh.php, 里面的内容，你可以在 @appfront/theme/terry/theme01/里面新建文件 mailer/customer/account/forgotpassword/body\_zh.php（如果存在，则不需要创建） 然后将上面的文件内容复制到新建的文件中，进行修改即可，这样就完成了邮件内容的添加或者重写。

通过以上四种方式的翻译，我们就完成了整个fecshop多语言翻译的闭环。

注意： 如果您不添加翻译，则默认就会使用英语的翻译。

关于邮件更详细的介绍，可以参看：[Fecshop 邮件](http://www.fecshop.com/doc/fecshop-guide/instructions/cn-1.0/fecshop_email.php)

## 三：各个入口的语言配置

当我们在上面的步骤中设置了语言，以及各个语言对应的翻译文件后， 我们需要为各个入口设置对应的默认语言

### 设置各个域名下面的默认语言

打开 @app/config/fecshop\_local\_services/Store.php

<?php

return [

'store' => [

'class' => 'fecshop\services\Store',

'stores' => [

// store key：域名去掉http部分，作为key，这个必须这样定义。

'fecshop.appfront.fancyecommerce.com' => [

'language' => 'en\_US', // 语言简码需要在@common/config/fecshop\_local\_services/FecshopLang.php 中定义。

'languageName' => 'English', // 语言简码对应的文字名称，将会出现在语言切换列表中显示。

'localThemeDir' => '@appfront/theme/terry/theme01', // 设置当前store对应的模板路径。关于多模板的方面的知识，您可以参看fecshop多模板的知识。

'thirdThemeDir' => [], // 第三方模板路径，数组，可以多个路径

'currency' => 'USD', // 当前store的默认货币,这个货币简码，必须在货币配置中配置

/\*

\* 当设备满足什么条件的时候，进行跳转。

\* 这种方式不怎么高效，最好的方式是在nginx中配置。

\*/

...

设置：

language ：第一部分中， @common/config/fecshop\_local\_services/FecshopLang.php 中进行配置语言项， 这里填写的值，必须该配置文件中出现。

languageName 自定义语言label，这个会在前端显示出来具体的名字（因为简码，用户 是看不明白的），譬如在appfront端的顶部左侧显示的多语言切换列表，apphtml5端 首页底部显示的多语言切换列表

然后就可以访问查看了，当把该入口的各个域名对应的默认语言设置好后， 访问这个语言，就可以看到相应的语言了。

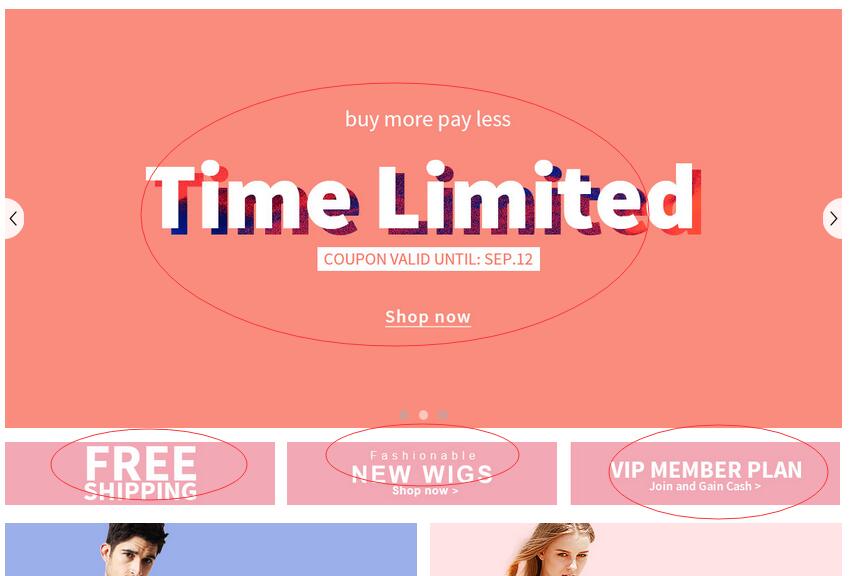
## 四：如何添加一门新语言

参看文档：<http://www.fecshop.com/topic/616>

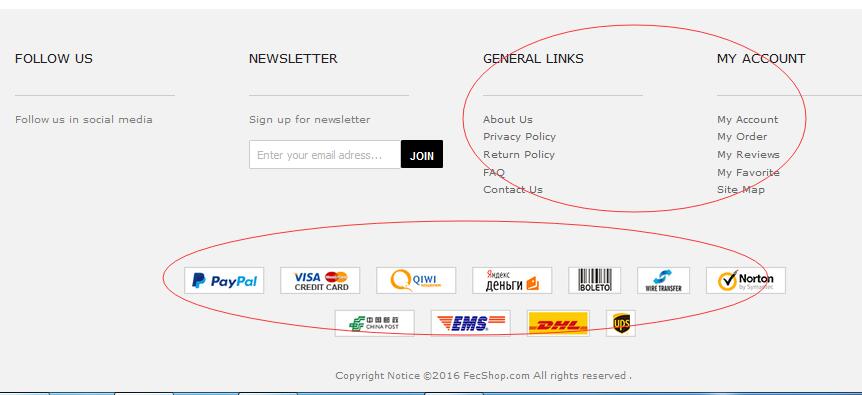
# Fecshop 静态块

fecshop 静态块是指一块静态的html段，可以在后台通过配置的方式进行修改， 这样，可以很方面的让工作人员进行修改。

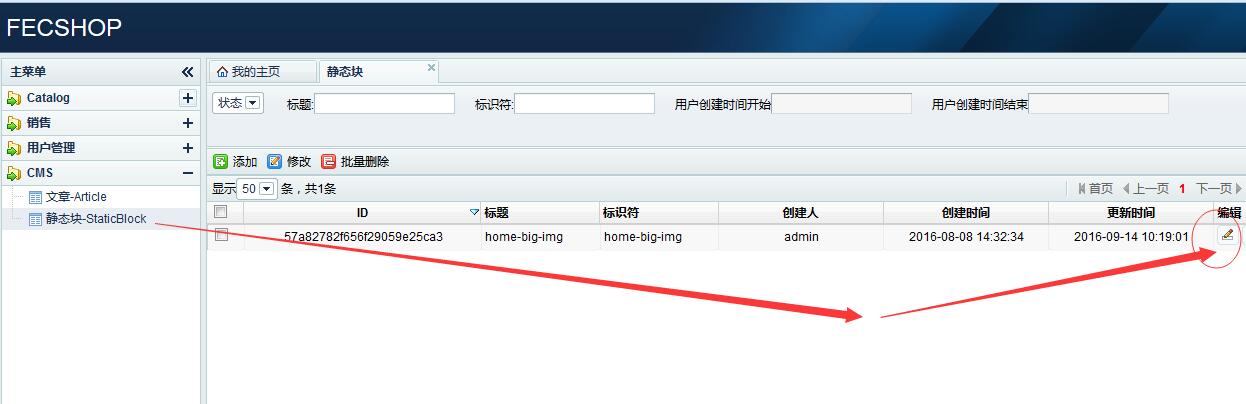
譬如首页的轮换大图，

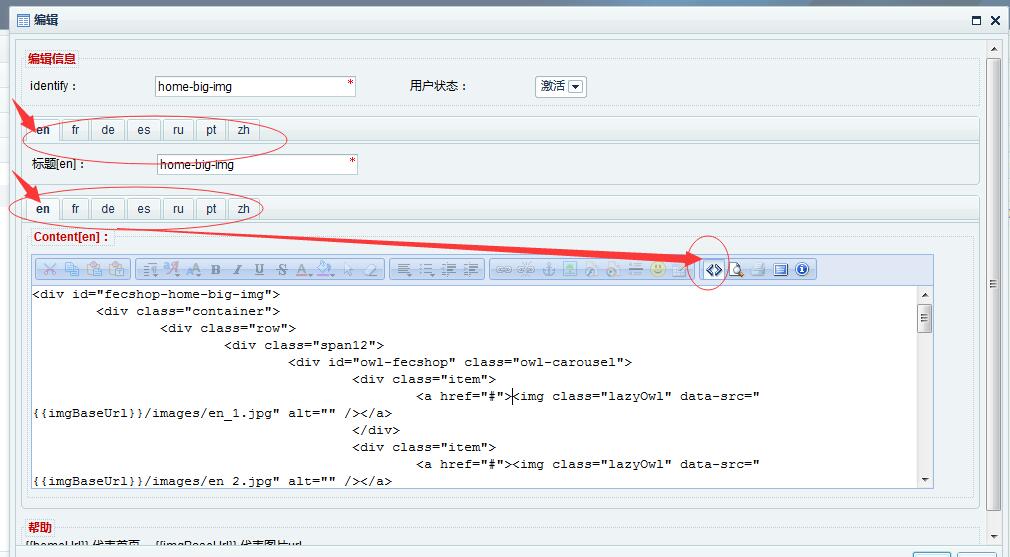


页面底部的文件条款等区块



您可以在后台进行编辑





在content（内容部分），{{homeUrl}} 代表首页。 {{imgBaseUrl}} 代表图片url。 其中，{{imgBaseUrl}}为 对应的文件路径为@appimage/app入口名/

### 使用方法

1.首先在后台新建一个static block，譬如key为home-big-img

2.在代码中调用

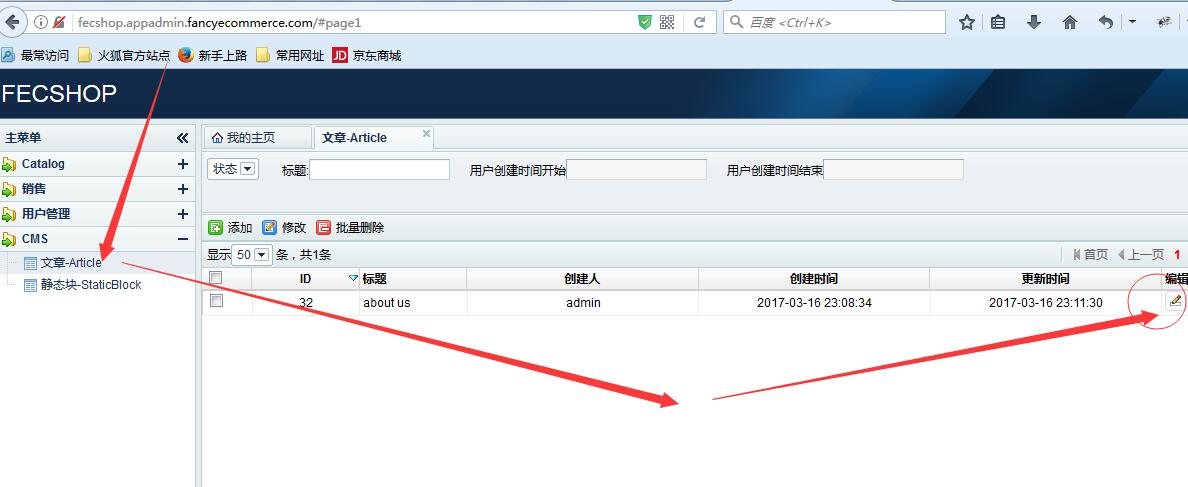
Yii::$service->cms->staticblock->getStoreContentByIdentify('home-big-img','appfront')

第一个参数为key，第二个参数为入口的名字，这个参数的作用 是为了生成url（域名部分）

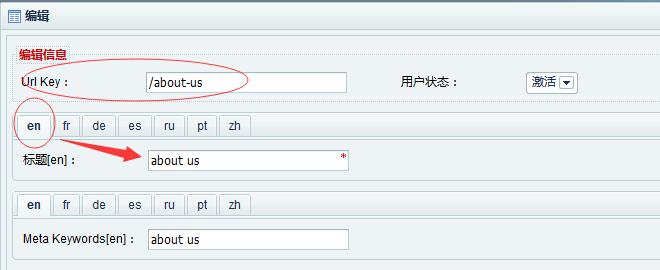
# Fecshop page页（文章）

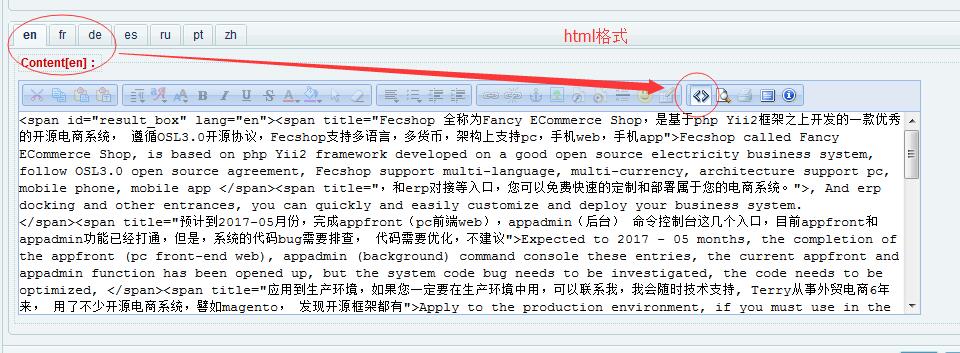
fecshop page,指的是一些单页面，一般用来写文字条款，譬如http://fecshop.appfront.fancyecommerce.com/index.php/about-us

您可以在后台编辑



弹出编辑框





然后保存，保存后，查看生成的url key就是url key，当然您也可以自定义url key。 用url key放到域名后面就可以访问了，譬如我新建的关于我们的 url key为 about-us， 那么我访问 <http://fecshop.appfront.fancyecommerce.com/index.php/about-us> 即可访问我新建的page页。

通过page，可以快速的新建一些文字页面。

# fecshop 首页

fecshop 首页，也就是home页，属于进入网站的主页面

### 1.配置

打开配置文件 @fecshop/app/appfront/config/appfront.php ，可以看到如下的代码：

// 首页对应的url key

'urlManager' => [

'rules' => [

'' => 'cms/home/index',

],

],

这是Yii2的rules配置，配置 '' => 'cms/home/index',， 这个配置代表的是首页地址对应 cms/home/index， 也就是cms模块下面的HomeController里面的actionIndex()方法， 也就是文件 @fecshop\app\appfront\modules\Cms\controllers\HomeController.php 里面的actionIndex()方法，

对于Yii2 controller的知识，可以参看：<http://www.yiichina.com/doc/guide/2.0/structure-controllers>

对于Yii2模块，可以参看：<http://www.yiichina.com/doc/guide/2.0/structure-modules>

url routing 可以参看：<http://www.yiichina.com/doc/guide/2.0/runtime-routing>

# Fecshop 404页

当一些url，找不到相应的controller，就会进入404页面，状态码为404，

404页面，使用的是Yii的 errorHandler组件，fecshop在配置文件 @fecshop/app/appfront/config/appfront.php中进行的配置。

'errorHandler' => [

'errorAction' => 'site/helper/error',

],

模板路径为：@fecshop/app/appfront/theme/base/front/site/helper/error.php

如果您想重写404页面，通过多模板重写的方式即可。

# Fecshop 小部件

fecshop小部件，不同于fecshop的小部件，fecshop的小部件是集数据库数据与view文件 生成的一个区块，譬如网站侧栏的热销产品，做成一个小部件，可以很方便的 在其他页面进行引入。当然，fecshop的head header（头部），footer（尾部）都是小部件实现的。

### 小部件结构：

小部件由两部分组成，动态数据提供部分 和 提供html代码的view部分，共同组合 ，显示出来小部件。

### 小部件配置：

@fecshop/config/services/Page.php里面进行小部件的配置，示例代码：

'widget' => [

'class' => 'fecshop\services\page\Widget',

'widgetConfig' => [

'head' => [

# 动态数据提供部分

'class' => 'fecshop\app\appfront\widgets\Head',

# 根据多模板的优先级，依次去模板找查找该文件，直到找到这个文件。

'view' => 'widgets/head.php',

# 缓存

'cache' => [

'enable' => false, # 是否开启

'timeout' => 4500, # 缓存过期时间

],

],

'header' => [

'class' => 'fecshop\app\appfront\widgets\Headers',

# 根据多模板的优先级，依次去模板找查找该文件，直到找到这个文件。

'view' => 'widgets/header.php',

'cache' => [

'enable' => false,

'timeout' => 4500,

],

],

'topsearch' => [

'view' => 'widgets/topsearch.php',

],

譬如header小部件，数据提供部分为:@fecshop\app\appfront\widgets\Headers.php, view部分为：@fecshop\app\appfront\theme\base\default\widgets\header.php。

其中，head header都是一个小部件，里面的class是动态数据提供部分， view是显示部分，cache是缓存，您可以通过该配置将缓存打开，以及设置缓存的超时时间。

小部件的view部分遵循fecshop的多模板机制，可以在二开theme路径下添加 相应文件实现view部分的重写，可以通过classMap机制进行动态数据提供class文件 的重写。

### 小部件的使用：

<?= Yii::$service->page->widget->render('header',$this); ?>

上面传的的$this，在view文件中对应的变量为：$parentThis，举例子可以参看head小部件

<?= Yii::$service->page->widget->render('head',$this); ?>

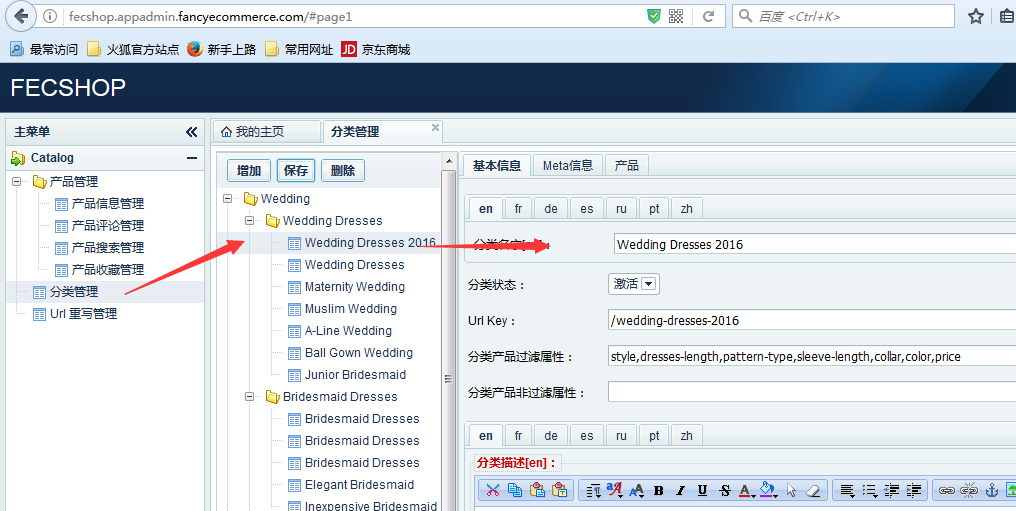
<?php $parentThis->head() ?>

$parentThis就是render()函数调用传入的$this变量。

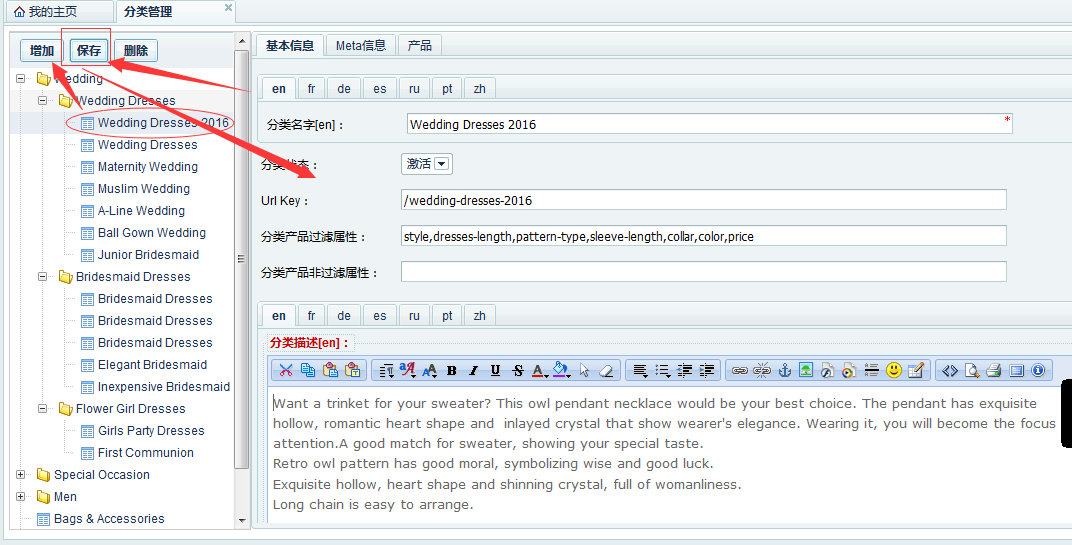
# Fecshop 分类

这里说的分类，就是在前台菜单导航栏里面的内容，包含home，产品分类，自定义分类等。

### 产品分类，自定义分类等

1.您可以通过下面的方式查看分类信息 

2.如果您想增加一个子分类，那么首先点击父分类，然后点击增加，在刷新后的右侧， 填写分类的信息，然后保存即可完成。



3.如果您想给分类添加产品可以按照下图所示，勾选产品。 

全部产品检索按钮：点击后，列表中可以查看所有的产品（已经勾选过的产品， 上面会有勾选的标志），您可以为这个分类勾选相应的产品

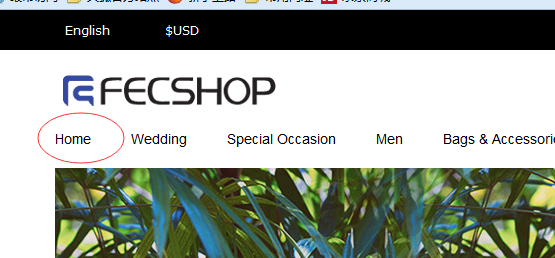
当前分类检索按钮：点击后，只显示当前分类已经勾选了的产品。

4.如果您新建了产品分类，但是没有勾选产品，那么在产品编辑的地方，也可以勾选产品。 勾选后，在分类部分，可以看到对应的产品。

5.新建分类后，在前端展示分类部分会显示新建的分类。

### HOME部分：

HOME部分，也就是在菜单，图片如下：



您可以通过配置的方式来配置是否显示home，配置的文件为： @appfront/config/fecshop\_local\_services/Page.php

'menu' => [

'displayHome' => [

'enable' => true, # 是否在菜单中显示home

'display'=> 'Home', # 显示对应的字符。

],

### 自定义菜单部分

对于fecshop，菜单的主体是产品分类菜单，其他的归结为自定义菜单。

自定义的配置的文件为： @appfront/config/fecshop\_local\_services/Page.php

/\*\*

\* 在菜单前面部分（产品分类菜单的前面部分）的自定义菜单。参考behindCustomMenu里面的格式

\*/

'frontCustomMenu' => [

],

/\*\*

\* 在菜单后面部分（产品分类菜单的前面部分）的自定义菜单

\*/

'behindCustomMenu' => [

[

'name' => 'custom menu', # 菜单名字

'urlPath' => '/my-custom-menu.html', # 菜单对应的url

'childMenu' => [ # 子菜单

[

'name' => 'my custom menu 2',

'urlPath' => '/my-custom-menu-2.html',

],

[

'name' => 'my custom menu 3',

'urlPath' => '/my-custom-menu-2.html',

'childMenu' => [

[

'name' => 'my custom menu 3',

'urlPath' => '/my-custom-menu-2.html',

],

[

'name' => 'my custom menu 3',

'urlPath' => '/my-custom-menu-2.html',

],

],

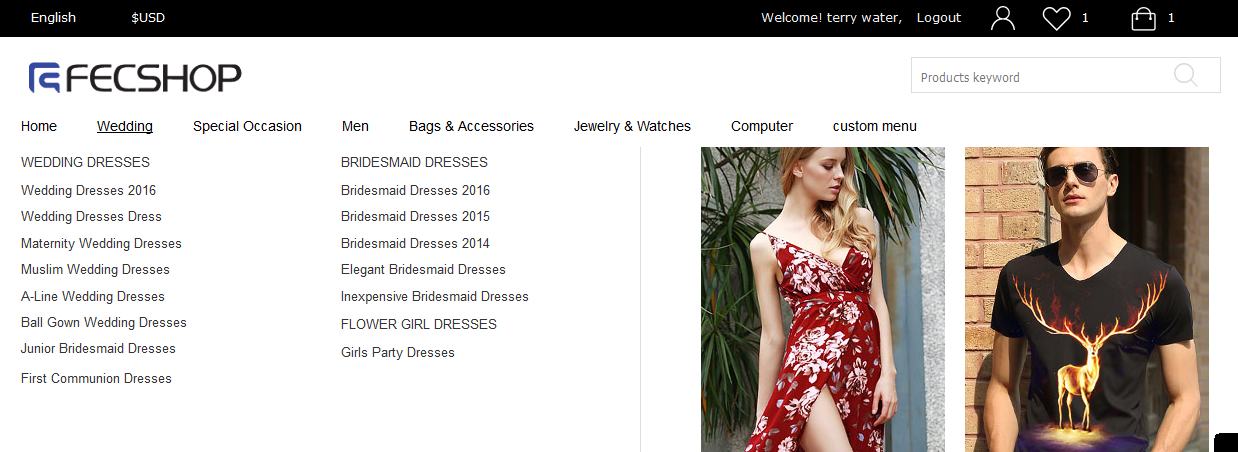
],

],

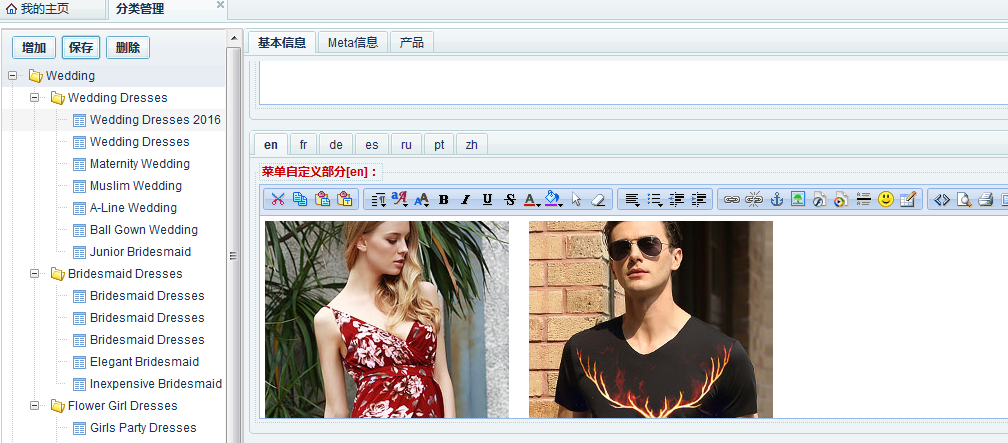
],

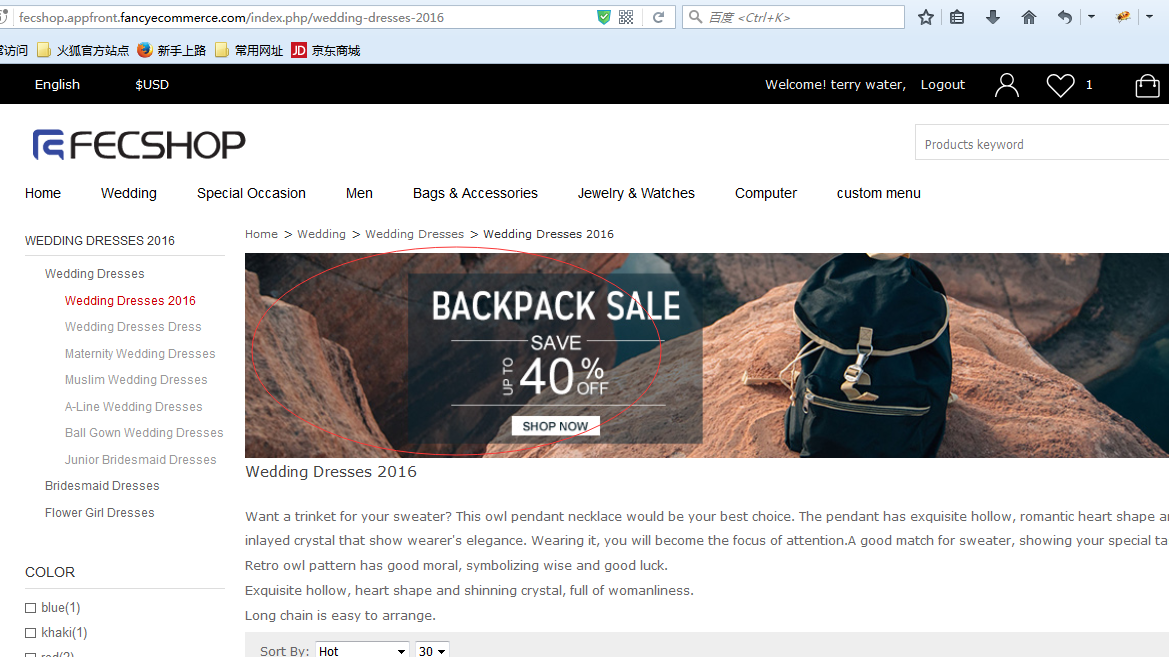
],

在上面自定义配置了菜单，就会在菜单导航栏中显示出来。

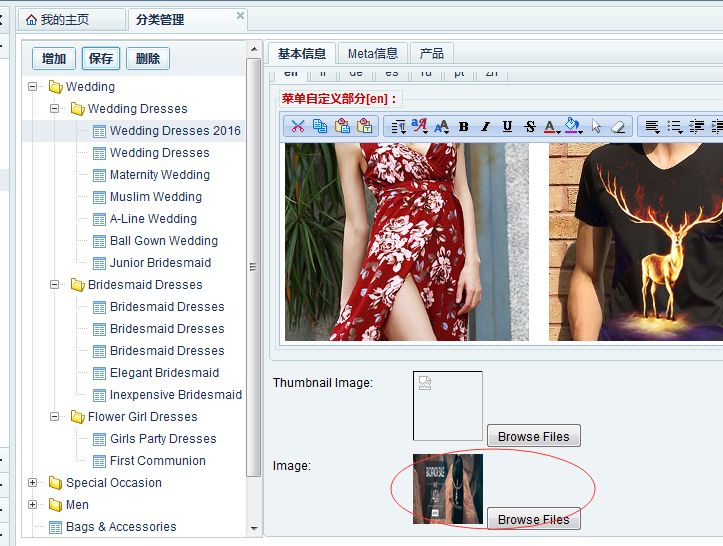


另外，需要说明一下分类编辑里面的内容，对于一级分类弹出部分右侧的图片内容 ，是在分类这里进行编辑的：

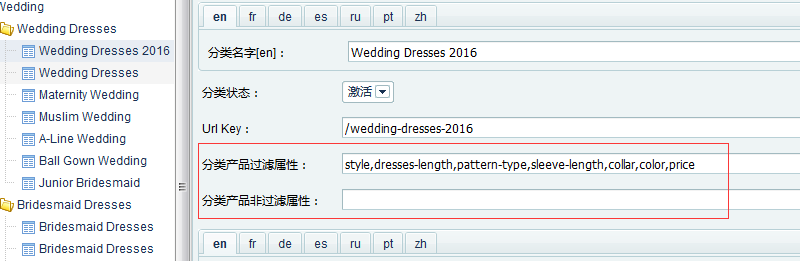


对于下图中，点击分类进如分类详细页面的图片，如图： 

是在这里编辑的，如图：



另外：

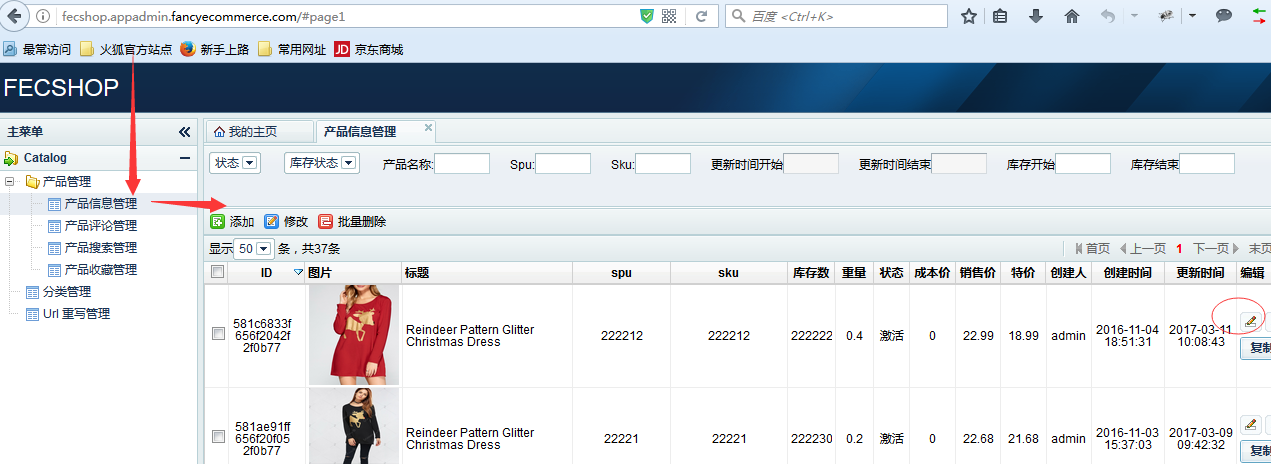


这里是分类中侧栏的产品过滤属性，这个在产品中详细解说。

# Fecshop 产品

作为电商系统，很多地方都是围绕产品展开的，譬如产品分类，产品搜索，产品评论， 产品收藏，购物车产品，订单产品等等。

### 后台产品编辑



### 产品属性组：

产品除了价格，名字，库存等一些公用的属性外， 产品根据不同的分类，会有一些独有的属性， 譬如电脑类产品会有cpu型号，主板型号等属性， 衣服类有颜色尺码等属性，在这里引入产品属性组的概念， 一类产品从属于一个属性组。

属性组的配置：

打开文件：@common/config/fecshop\_local\_services/Product.php文件 可以看到对属性组的配置

配置文件的内容太多，这么不贴出来的，您可以打开这个文件查看具体内容，里面有注释

属性组里面的属性有类别，目前有三个类：

1、**spu\_attr**

1.1 spu和sku的概念：sku是产品的唯一标示，最小库存单位，譬如一款鞋子为一个spu，这款鞋子的不同的颜色和不同的尺码为一个sku，

1.2 什么是spu属性（spu\_attr），当一款鞋子的不同的sku是颜色和尺码的不同，那么颜色和尺码就是这款鞋子的spu属性（spu\_attr）。

1.3 spu\_attr是用来在产品详细页面，将这个spu下面的所有sku串联起来，譬如页面：<http://fecshop.appfront.fancyecommerce.com/index.php/raglan-sleeves-letter-printed-crew-neck-sweatshirt-53386451-77774122> 访问这款衣服的每一个SKU，都会把其他的sku列出来，点击后，就进去其他sku的页面，您会发现url是变化的，这个是类似于京东的方式。 注意：

1.3.1 因为组合的复杂性，spu attr最多设置2个属性，不能超过2个(如果超过会出问题。spu属性会都不显示)，超过两个请使用customer\_option的方式。也就是类似于淘宝的方式。

1.3.2 这里的属性需要严格按照下面的格式进行配置，如果您想配置自己的spu属性，那么

1.3.3 通过链接：<https://fecshop.appfront.fancyecommerce.com/raglan-sleeves-letter-printed-crew-neck-sweatshirt-41214715>您可以看到，color属性是用图片来显示的， 在spu\_attr中，第一个属性默认用图片显示，譬如：

'spu\_attr' => [ # spu用来区分sku的属性，譬如下面的属性的不同，对应不同的sku，进而是不同的库存

# 第一个属性会被用户当做图片来显示。

'color' => [

'dbtype'=> 'String',

'label' =>'颜色',

'name' =>'color',

'sort\_order' => 1,

'display'=>[

'type' => 'select',

'data' => [

'one-color' => 'one-color',

'red' => 'red',

'white' => 'white',

'black' => 'black',

'blue' => 'blue',

'green' => 'green',

'yellow' => 'yellow',

'gray' => 'gray',

'khaki' => 'khaki',

'ivory' => 'ivory',

'beige' => 'beige',

'orange' => 'orange',

'cyan' => 'cyan',

'leopard' => 'leopard',

'camouflage' => 'camouflage',

'silver' => 'silver',

'pink' => 'pink',

'purple' => 'purple',

'brown' => 'brown',

'golden' => 'golden',

'multicolor' => 'multicolor',

'white & blue' => 'White & Blue',

'white & black' => 'White & Black',

]

],

//'require' => 0,

//'default' => 2,

],

# 第二个属性不会当做图片来显示

'size' => [

'dbtype'=> 'String',

'label' =>'尺码',

'name' =>'size',

'sort\_order' => 2,

'display' =>[

'type' =>'select',

'data' =>[

'one-size' => 'one-size',

'S' => 'S',

'M' => 'M',

'L' => 'L',

'XL' => 'XL',

'XXL' => 'XXL',

'XXXL' => 'XXXL',

]

],

//'require' => 0,

//'default' => 2,

],

],

color 将会被当做图片来显示出来。

2、**general\_attr（普通属性）**，可以是各种格式的值，譬如时间格式，email格式，下拉条选择值等。这些属性可以用于分类侧栏属性过滤。

示例配置：

'general\_attr' => [

# 这是input type='text' 的类型

'my\_remark' => [

'dbtype'=> 'String',

'label'=>'我的备注',

'name'=>'my\_remark',

'display'=>[

'type' => 'inputString', # 字符串格式的属性

],

'require' => 0,

],

# 这是input type='email' 的类型

'my\_email' =>[

'dbtype'=> 'String',

'label'=>'我的邮箱',

'name'=>'my\_email',

'require' => 0,

'display'=>[

'type' => 'inputEmail', # 字符串格式的属性（email格式验证）

],

],

# 这是input type='date' 的类型

'my\_date' => [

'label'=>'我的日期',

'name'=>'my\_date',

'display'=>[

'type' => 'inputDate', # 字符串格式的属性（Date格式验证）

],

],

# 这是<select> 的类型

'style' => [

'dbtype'=> 'String',

'label' =>'类型',

'name' =>'style',

'display' =>[

'type' =>'select', # 下拉条选择格式的属性

'data' =>[

'Casual' => 'Casual',

'Cute' => 'Cute',

'Sexy & Club'=> 'Sexy & Club',

'Bohemian' => 'Bohemian',

'Vintage ' => 'Vintage ',

'Brief' => 'Brief',

'Work' => 'Work',

'Novelty' => 'Novelty',

]

],

],

'dresses-length' => [

'dbtype'=> 'String',

'label' =>'裙长',

'name' =>'dresses-length',

'display' =>[

'type' =>'select', # 下拉条选择格式的属性

'data' =>[

'Mini' => 'Mini',

'Knee-Length' => 'Knee-Length',

'Mid-Calf'=> 'Mid-Calf',

'Ankle-Length' => 'Ankle-Length',

'Floor-Length ' => 'Floor-Length ',

]

],

],

'pattern-type' => [

'dbtype'=> 'String',

'label' =>'款式',

'name' =>'pattern-type', # 属性名字

'display' =>[

'type' =>'select', # 下拉条选择格式的属性

'data' =>[

'Animal' => 'Animal',

'Character' => 'Character',

'Floral'=> 'Floral',

'Geometric ' => 'Geometric ',

'Leopard '=> 'Leopard ',

'Letter'=> 'Letter',

'Paisley'=> 'Paisley',

'Patchwork'=> 'Patchwork',

'Polka Dot'=> 'Polka Dot',

'Print'=> 'Print',

'Striped'=> 'Striped',

]

],

],

'sleeve-length' => [

'dbtype'=> 'String',

'label' =>'袖长',

'name' =>'sleeve-length',

'display' =>[

'type' =>'select',

'data' =>[

'Sleeveless' => 'Sleeveless',

'Short-Sleeves' => 'Short Sleeves',

'Half-Sleeves'=> 'Half Sleeves',

'3-4-Length-Sleeves ' => '3/4 Length Sleeves ',

'Long-Sleeves '=> 'Long Sleeves ',

]

],

],

'collar' => [

'dbtype'=> 'String',

'label' =>'领口', # 后台显示的中文名（目前后台只有中文）

'name' =>'collar',

'display' =>[

'type' =>'select',

'data' =>[

'Round Neck' => 'Round Neck', # 下拉条里面对应的各个可以选择的值。

'V-Neck' => 'V-Neck',

'Hooded' => 'Hooded',

'Turn-down-Collar' => 'Turn-down Collar',

]

],

//'require' => 0,

//'default' => 2,

],

],

3、**custom\_options用户自定义属性**，显示方式方面有点和spu属性类似，spu属性显示的方式是京东的方式，点击每一个选项是url跳转的 用户自定义类似于淘宝的方式，选择各个颜色尺码，页面是不跳转的，各个颜色尺码有相应的图片，库存，价格，sku等。 您可以查看演示地址：http://fecshop.appfront.fancyecommerce.com/index.php/reindeer-pattern-glitter-christmas-dress-86519596 您可以给产品属性组添加的属性类型就上面几种，在后台编辑产品的时候，选择不同的属性组，就会加载相应的属性出来。

示例配置：

'custom\_options' => [

'my\_color' => [

'dbtype'=> 'String', #类型

'label' =>'My Color', # 显示的名字

'name' =>'color', # 在数据库中存在的列名

'showAsImg' => true, # （在前端展示部分）通过图片的方式展示属性。譬如；http://fecshop.appfront.fancyecommerce.com/index.php/reindeer-pattern-glitter-christmas-dress-86519596，

# 你会发现，该属性对应的显示方式不是值，而是产品的图片。

'require' => 1, # 1代表是必填选项，0代表选填

'display'=>[

'type' => 'select',

'data' => [

'red' => 'red',

'white' => 'white',

'black' => 'black',

'blue' => 'blue',

'green' => 'green',

'yellow' => 'yellow',

'gray' => 'gray',

'khaki' => 'khaki',

'ivory' => 'ivory',

'beige' => 'beige',

'orange' => 'orange',

'cyan' => 'cyan',

'leopard' => 'leopard',

'camouflage' => 'camouflage',

'silver' => 'silver',

'pink' => 'pink',

'purple' => 'purple',

'brown' => 'brown',

'golden' => 'golden',

'leopard' => 'leopard',

'multicolor' => 'multicolor',

'white & blue' => 'White & Blue',

'white & black' => 'White & Black',

]

],

],

'my\_size' => [

'dbtype'=> 'String',

'label' =>'My Size',

'name' =>'size',

'require' => 1,

'display' =>[

'type' =>'select',

'data' =>[

'S' => 'S',

'M' => 'M',

'L' => 'L',

'XL' => 'XL',

'XXL' => 'XXL',

'XXXL' => 'XXXL',

]

],

],

],

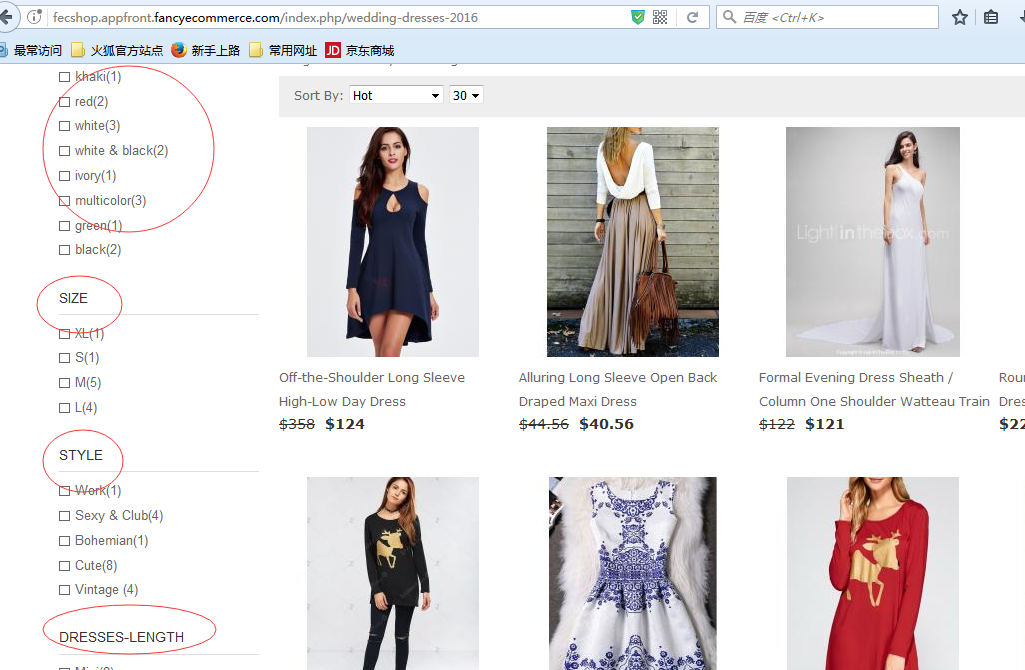
通过链接：<https://fecshop.appfront.fancyecommerce.com/reindeer-pattern-glitter-christmas-dress> 。您可以看到颜色是用图片实现的，如果您想用图片显示，那么 您需要设置'showAsImg' => true 即可。 只能有一个custom\_option属性设置为true，如果多个设置为true，只有一个 有效，其他的会被忽略。

上面是属性组里面的三种属性类型，分别是： spu\_attr general\_attr 和 custom\_options ，我们可以通过配置的方式添加多个属性组，每一个属性组包含上面 的三种属性，譬如衣服裙子一个属性组，电脑一个属性组，手机一个属性组 等等， 通过属性组配置的方式可以很方便的扩展产品独有的属性， 在配置文件中配置好产品属性组后， 在后台 编辑产品的时候选择相应的属性组即可添加相应的属性，对于属性组里面的属性，一旦编辑后 。尽量不要经常的改动。



### 分类页面产品属性过滤的设置

就是前端页面侧栏的产品过滤，如图：



如何设置呢？

1、全局设置：首先打开配置文件 @appfront/config/fecshop\_local\_modules/Catalog.php

配置

return [

'catalog' => [

'params'=> [

###############################

## category部分设置 ##

###############################

'category\_breadcrumbs' => false, # 是否显示分类的面包屑导航。

/\*\*

\* 注意：做侧栏分类产品过滤的属性，必须是select类型的，其他的类型请不要用，

\* 对于select类型，目前不支持多语言数据库存储，select类型的各个值是通过前端翻译文件来实现翻译的、

\* 对于color size 对应的保存值，只可以使用 '数字','字符','空格','&','-','\_' 这6类字符

\*/

'category\_filter\_attr' =>[

'color','size',

],

'category\_filter\_category' => true,

'category\_filter\_price' => true,

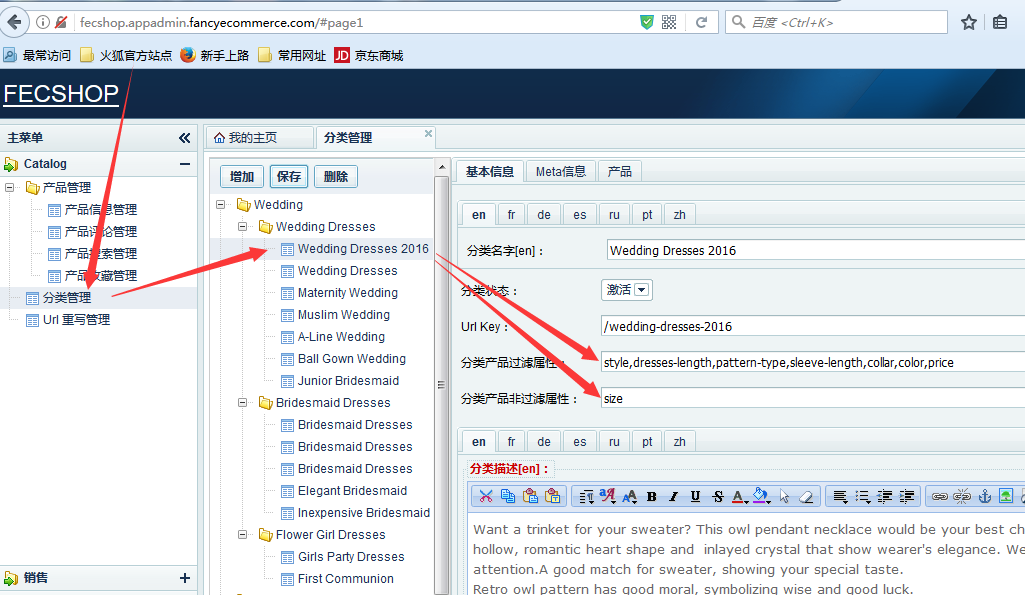
category\_filter\_attr 就是配置分类侧栏的产品属性过滤，目前设置了color和size ，这里设置后是一个公用的设置，也就是所有的分类就默认为这两个属性的过滤。

category\_filter\_category代表侧栏是否显示分类的过滤

category\_filter\_price代表侧栏是否显示价格的过滤。

2、个性化设置

在上面设置的全局设置的基础上，可以进行个性化设置，后台打开分类，如图：



**分类产品过滤属性：**是在全局设置的基础上添加一些新的产品属性，用于分类侧栏属性过滤

**分类产品非过滤属性：**是在全局设置的基础上删除一些产品属性，譬如下面的图中， 添加的是size，虽然在全局设置中添加了size，但是该分类添加了size，因此该 分类侧栏的属性过滤不会有size。

通过上面，我们可以很方便的添加产品在分类侧栏的属性过滤。

### 搜索页面产品属性过滤的设置

1.首先设置搜索语言，@common/config/fecshop\_local\_services/Search.php

1.1

return [

'search' => [

'filterAttr' => [

'color','size', # 在搜索页面侧栏的搜索过滤属性字段

],

'childService' => [

'mongoSearch' => [

'searchIndexConfig' => [ #设置用于全文搜索的产品属性以及权重，权重高的属性，排名靠前。

'name' => 10, # 产品name作为full search text的属性，权重为10

'description' => 5, # 产品description作为full search text的属性，权重为5

],

# more: https://docs.mongodb.com/manual/reference/text-search-languages/#text-search-languages

'searchLang' => [

'en' => 'english',

'fr' => 'french',

'de' => 'german',

'es' => 'spanish',

'ru' => 'russian',

'pt' => 'portuguese',

],

],

'xunSearch' => [

'fuzzy' => true, # 是否开启模糊查询

'synonyms' => true, #是否开启同义词翻译

'searchLang' => ['zh'],

],

],

]

];

1.1 filterAttr是添加在搜索页面侧栏用于过滤的产品属性。

1.2 searchIndexConfig是设置用于全文搜索的产品属性以及权重，权重高的属性，排名靠前。

1.3 searchLang是设置搜索的语言，全文搜索是需要根据语言切词的， 因此各个语言的搜索会略有差异，根据各个语言，生成对应的产品语言搜索表。

2.生成搜索表

首先需要根据上面的配置文件，设置您的所有的语言。也就是上面的searchLang。

每一个语言对应一个产品搜索表，这是因为mongodb的full text search，一个表只能有一个 语言选项，因此需要生成多个产品搜索表，生成的脚本为：

@fecshop/shell/search/fullSearchSync.sh，您进入@fecshop/shell/search下 执行sh fullSearchSync.sh即可。

### 设置产品的自定义属性

产品的自定义属性如图：



#### 1.产品的自定义有方式，京东模式和淘宝模式

1.1京东模式：类似于京东的产品，选择每一个选项后，页面会跳转到另外一个url页面， 也就是说：每一个选项就是一个产品（在数据库产品表中是一行数据）。

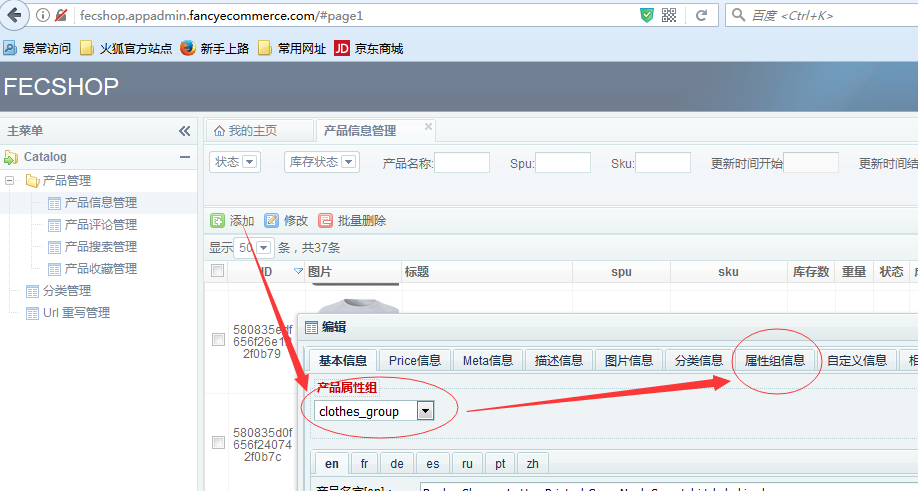
1.2淘宝模式：类似于淘宝商城的产品，选择一个选项后，产品页面不会跳转。

#### 2.下面详细说明：

2.1京东模式：这种模式，需要引入sku和spu的概念，关于这个概念在上面已经说明。 譬如页面（演示demo）：<http://fecshop.appfront.fancyecommerce.com/index.php/raglan-sleeves-letter-printed-crew-neck-sweatshirt-53386451-77774122>

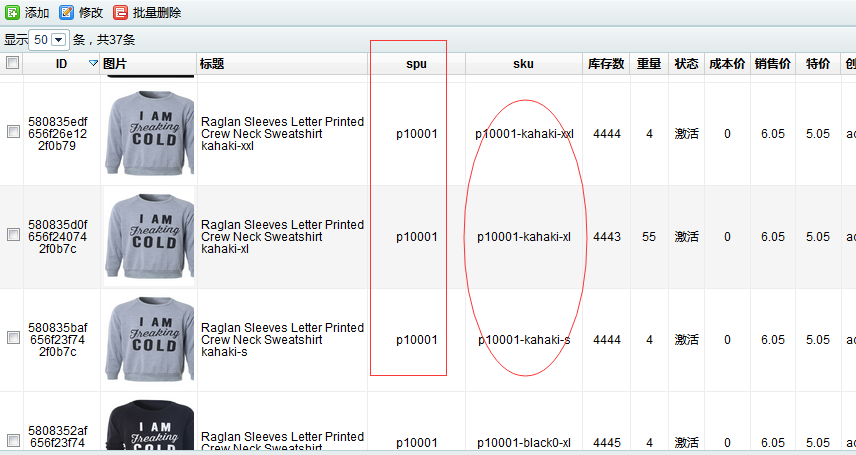
2.1.1配置：首先需要设置一个属性组，上面的内容已经说明，在配置文件中添加， 需要注意的spu属性不能超过两个（spu属性组合的复杂性）。

2.1.2新建产品，选择对应的属性组,如图



就可以查看到这个属性组中的属性，填写值即可。

填写sku和spu，需要注意的是，您新建的该款产品的spu必须相同，sku不同，一般用您的 spu+spu属性的值组合成sku，如图，我新建的spu对应的几个sku产品。



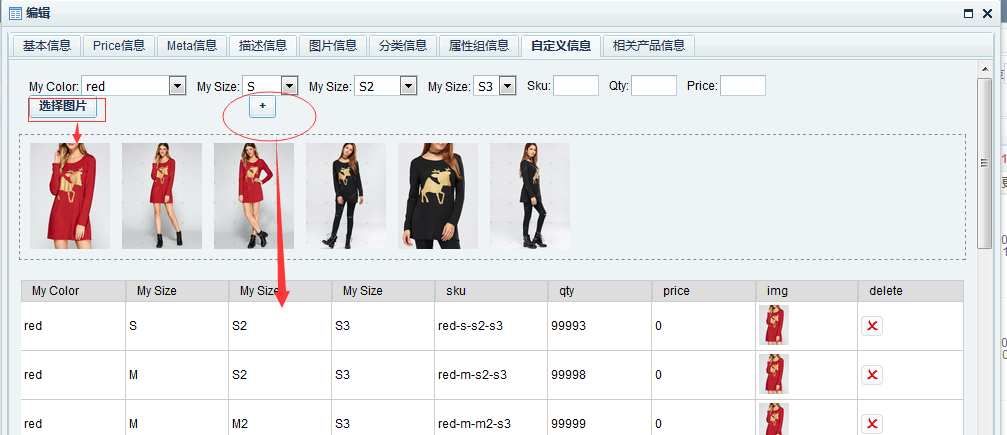
2.1.3 按照上面如图的方式配置完成后，基本就完成了配置，在前台就可以看到了。

2.2淘宝模式，演示demo：<http://fecshop.appfront.fancyecommerce.com/index.php/reindeer-pattern-glitter-christmas-dress-86519596>

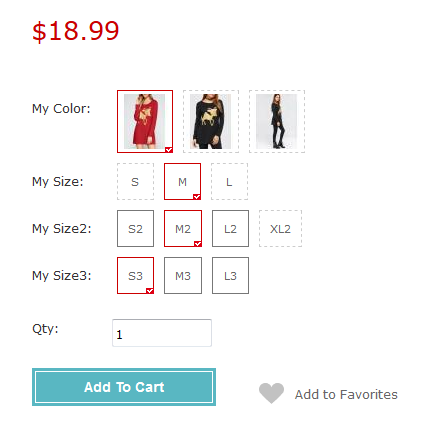
2.2.1配置：需要在属性组里面配置 custom\_options用户自定义属性，这个在上面已经 说明如何配置，您可以参考上面的说明中的文件中的例子进行修改。

2.2.2后台编辑

在属性组中添加了 custom\_options用户自定义属性，在新建产品的时候点击添加，弹出产品编辑的弹框， 选择相应的属性组，然后点击 “自定义信息”，如图：

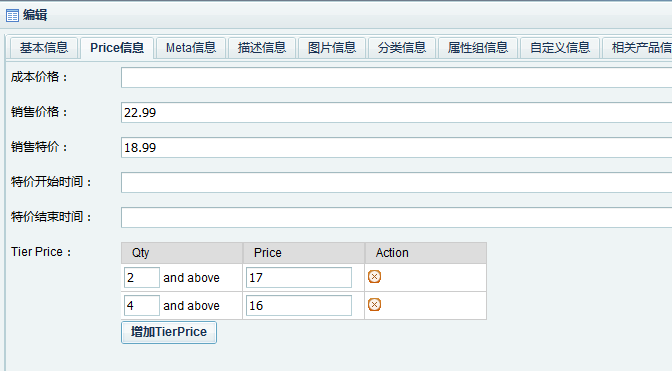


勾选，填写各个属性的值，以及选择图片，最后点击+号图标，就可以添加一行， 添加完了，保存，就可以到产品详细页面看了，譬如产品 http://fecshop.appfront.fancyecommerce.com/index.php/reindeer-pattern-glitter-christmas-dress-86519596



如果某些组合，您没有添加，就会看到某些选项是虚线标示，如果没有库存，也会虚线标示。 对于该产品，产品sku的库存会无效，产品自定义部分的库存才是相应选项的库存 ，当该产品被下单后，也是在该选项对应的qty部分减少产品的库存，而不是sku的库存（qty）

### 产品价格



成本价格：代表产品的成本，也就是采购在fecshop中，只是记录一下，没有其他用处， 当然，您可以通过脚本，生成您的出售价格。

销售价格：也就是产品在页面显示的价格

销售特价：产品的特价，当设置了产品的特价，而且特价时间没有过期，则特价是有效的， 当销售特价有效时，销售价格就会失效。

特价开始时间：产品销售特价开始的时间

特检结束时间：产品销售特价结束的时间

Tier Price： 批发价格

规定产品购买几个以上的价格。譬如购物2个以上17美元，购买4个以上16美元。

对于在产品分类页面显示的产品的几个都是一个产品的价格。产品中有一个最终价格， 是通过脚本计算出来的，这个最终价格用于在分类页面的价格排序以及价格过滤，该脚本一天跑一次。 文件为： @fecshop/shell/computeProductFinalPrice.sh， 可以在cron中配置该脚本一天跑一次。

### 产品导入和导出

目前后台还没有做这个功能。

# Fecshop 产品图片

fecshop 产品图片的实现思路，有很多参考了magento的图片生成的思路

### 1. 产品图片的存放路径

在配置文件 @fecshop/config/services/Product.php 中可以看到如下代码：

'image' => [

'class' => 'fecshop\services\product\Image',

'imageFloder' => 'media/catalog/product', # 产品图片存放路径。

//'allowImgType' => [ # 允许的图片类型

// 'image/jpeg',

// 'image/gif',

// 'image/png',

// 'image/jpg',

// 'image/pjpeg',

//],

'maxUploadMSize'=> 5, //MB # 图片最大尺寸

//'waterImg' => 'product\_water.jpg', # 水印图片

],

对于产品图片，是公用的，因此强制放到@appimage/common路径下面， 其中 imageFloder 的值代表的是相对路径，因此完整的图片存放路径为： @appimage/common/media/catalog/product 下面。

### 2. 产品图片的调用

原图的调用：

Yii::$service->image->getImgUrl('/2/01/20160905101021\_28071.jpg');

自定义尺寸的调用：

Yii::$service->product->image->getResize('/2/01/20160905101021\_28071.jpg',[230,230],false);

在自定义尺寸的调用的函数中 [230,230] 为width 和height的尺寸,详细参看函数：

/\*\*

\* $imgResize 可以为数组 [230,230] 代表生成的图片为230\*230，如果宽度或者高度不够，则会用白色填充

\* 如果 $imgResize设置为 230， 则宽度不变，高度按照原始图的比例计算出来。

\*/

protected function actionGetResize($imageVal, $imgResize, $isWatered = false)

{

$originImgPath = $this->getDir($imageVal);

if (!file\_exists($originImgPath)) {

$originImgPath = $this->getDir($this->defaultImg);

}

$waterImgPath = '';

if ($isWatered) {

$waterImgPath = $this->getDir('/'.$this->waterImg);

}

list($newPath, $newUrl) = $this->getProductNewPath($imageVal, $imgResize, $waterImgPath);

if (!file\_exists($newPath)) {

\fec\helpers\CImage::saveResizeMiddleWaterImg($originImgPath, $newPath, $imgResize, $waterImgPath);

}

return $newUrl;

}

图片按照尺寸调用的时候，php会去磁盘去查找该尺寸对应的产品图片 文件是否存在，如果不存在，php则会生成相应尺寸的图片存放到磁盘， 下次访问该图片，就直接调用生成后的图片。

上面的函数中,$isWatered代表是否打水印。

### 3.图片水印。

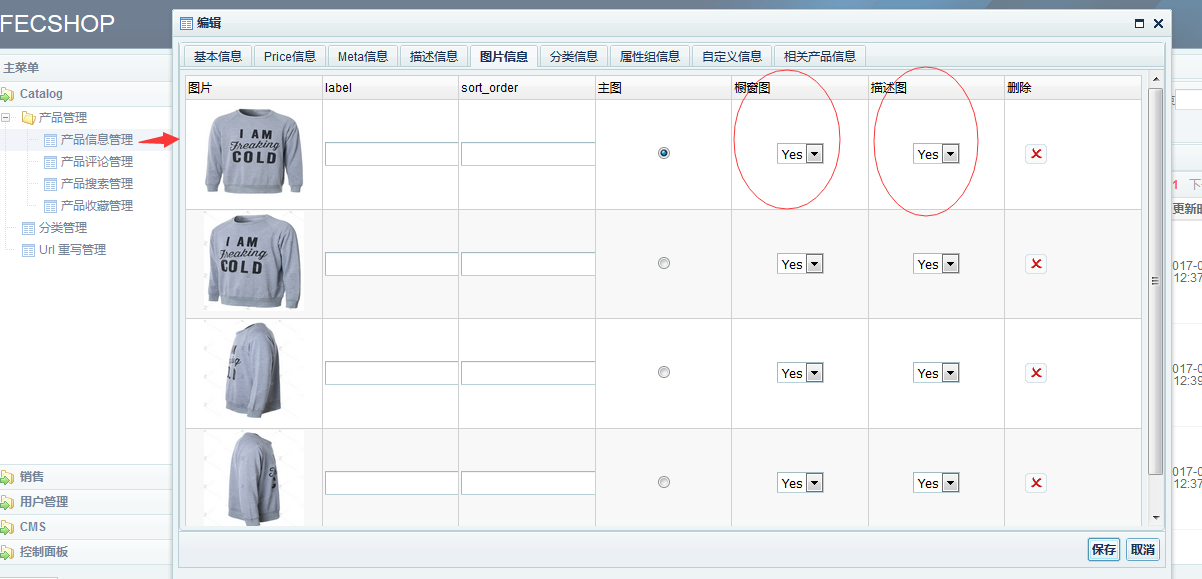
首先需要上传您的水印图。按照默认设置，水印图路径为 @appimage/common/media/catalog/product/product\_water.jpg ,您上传一张图覆盖这个水印图

然后在调用的时候， 自定义尺寸的调用：Yii::$service->product->image->getResize($product['image'],[230,230],true); ，第三个参数为true即可。

注意：目前原图是直接调用，如果想要打水印，您可以生成一个大尺寸的， 如果您不想让您的原图暴露，可以把程序里面所有的原图访问更改成 尺寸处理后的图片，然后nginx设置，封锁死原图访问url即可。

### 4.图片显示设置

在后台编辑的时候可以看到如下：

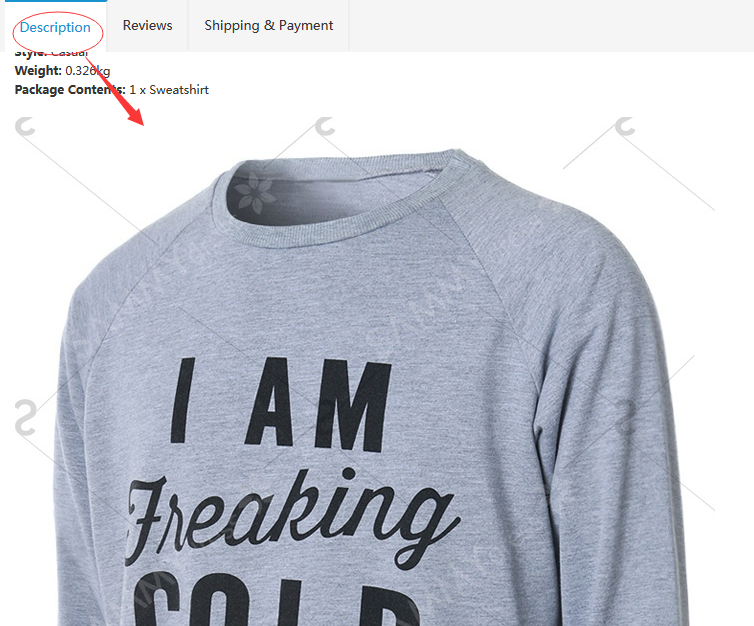


主图：代表产品的主图，在分类页面，购物车页面等，显示的都是产品的主图。

橱窗图：代表是的在产品详细页面顶部左侧显示的图片（如果是主图，无论是否设置成yes，都是橱窗图） 对于橱窗图的位置，参看图：



描述图：代表的是在产品详情页面，description描述下面显示的产品图



当一个产品设置成描述图，就会在产品描述部分下面显示出来。