MARTIN BROCHHAUS

ENTREPRENEUR, SOFTWARE ENGINEER, FREELANCE WEBDESIGNER,
HOUSE DJ, PHOTOGRAPHER NOOB, WORLD TRAVELER

29TH JUN 2011 | 5 NOTES

DJANGO-SHOP ADVENTURES: SIMPLE VARIATIONS

Today we are going to add django-shop-simplevariations to our little example project.

Luckily Chris Glass has already created a nice little app for this: **django-shop-simplevariations**.

During the last week I had a closer look at this and contributed some code which should make the installation and usage a breeze.

- First we need to add “shop_simplevariations” to our INSTALLED_APPS setting.
- Then we need to add “shop_simplevariations.cart_modifier.ProductOptionsModifier” to our SHOP_CART_MODIFIERS setting.
- Finally we need to catch the “/shop/cart/” URL in order to override the view class that would be used by django-shop with a new one that handles our variations. We can do this by modifying our urls.py like so:

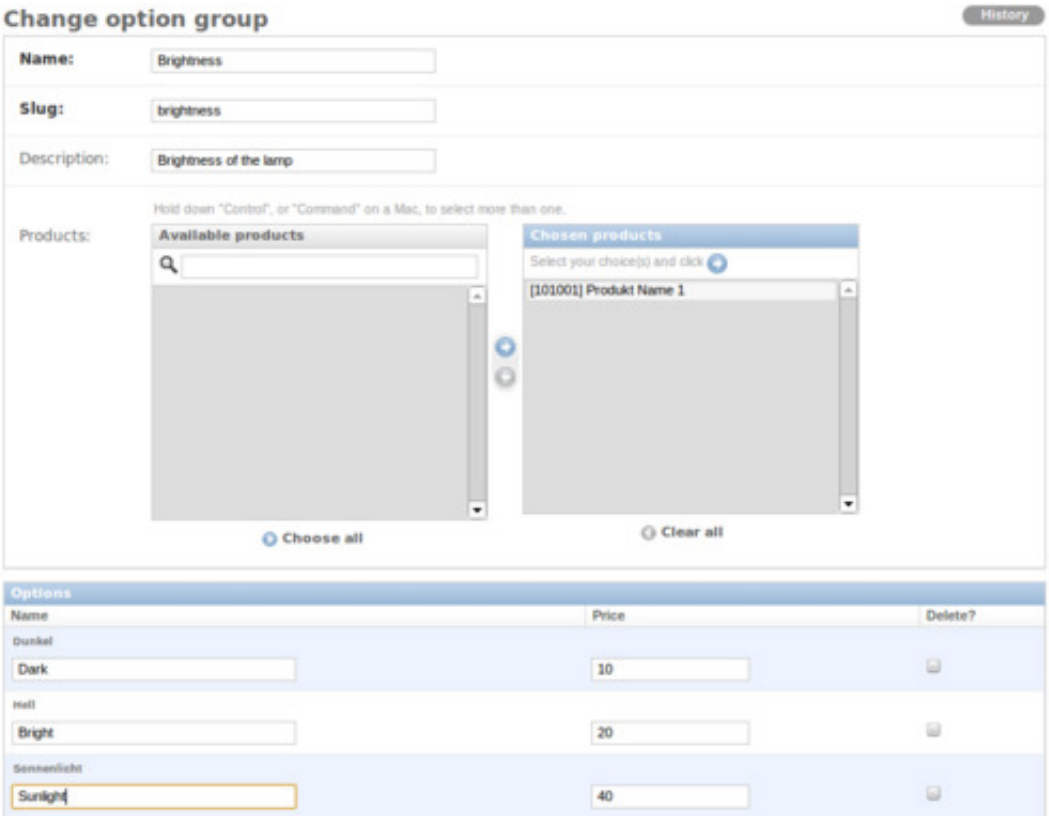
```
1  from django.conf.urls.defaults import *
2  from django.contrib import admin
3
4  from shop import urls as shop_urls
5  from shop_simplevariations import urls as simplevariations_urls
6
7
8  admin.autodiscover()
9
10
11 urlpatterns = patterns('',
12     (r'^admin/', include(admin.site.urls)),
13     (r'^shop/cart/', include(simplevariations_urls)),
```

```
14      (r'^shop/', include(shop_urls)),
15  )
```

gistfile1.py hosted with ❤ by GitHub view raw

All this is also documented in the README.txt of the django-shop-simplevariations repository.

This is nearly all we need to do. By now we should be able to go to our admin and add some OptionGroup objects:



Again we get a fancy widget for bulk assigning existing products to our OptionGroup. In this example I added a group called “Brightness” which has several options like “Dark”, “Bright” and “Sunlight”. Each option has a different price that will be added to the standard price of the product.

Finally we need to override the template “/shop/product_detail.html” and add drop down lists so that the customer is able to select the product variation that he prefers.

django-shop-simplevariations comes with a set of templatetags that make the creation of these drop down lists quite easy. Your code should look something like this:

```
1  {% load simplevariation_tags %}
2  <!-- Your product detail view here -->
3  <form method="post" action="{% url cart %}">{% csrf_token %}
4  {% with option_groups=object|get_option_groups %}
5    {% if option_groups %}
6      <div>
7        <h2>Variations:</h2>
```

```
      {% for option_group in option_groups %}
        <label for="add_item_option_group_{{ option_group.id }}">{{
        {% with options=option_group|get_options %}
          <select name="add_item_option_group_{{ option_group.id }}
            {% for option in options %}
              <option value="{{ option.id }}">{{ option.name }}</op
            {% endfor %}
          </select>
        {% endwith %}
      {% endfor %}
    </div>
  {% endif %}
{% endwith %}
<input type="hidden" name="add_item_id" value="{{object.id}}">
<input type="hidden" name="add_item_quantity" value="1">
<input type="submit" value="Add to cart">
</form>
```

gistfile1.py hosted with ❤ by GitHub view raw

That's it! When a customer buys your product with variations they will be treated as cart modifiers and will look like this (with the standard cart template that ships with the shop):

Your shopping cart

Product name	Unit price	Quantity	
Produkt Name 1	123	4	492
		Pink	40
		Dunkel	40
		Line Total:	572

In this case both variations ("Pink" and "Dunkel") cost 10 EUR, since we bought 4 products, each variation now makes up 40 EUR in the final bill.

As always, the updated code for this blog post can be found on **github**.

 **mbrochh** posted this

---

« Previous Next »
