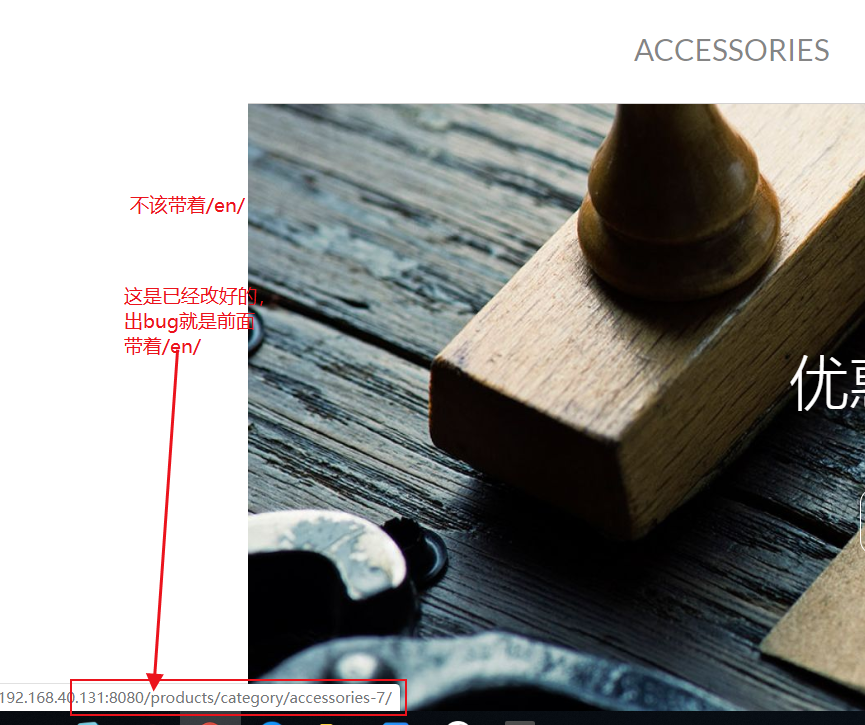
**Saleor任务**

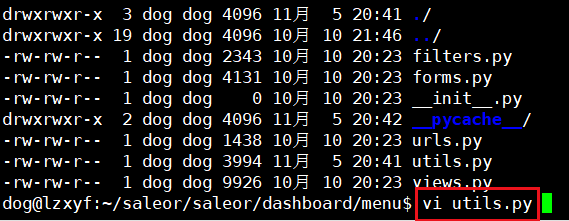
# 1 切换语言的下拉框（修改bug）



点击ACCESSORIES以后就无法更改语言。

## 问题解决：

### 修改位置：

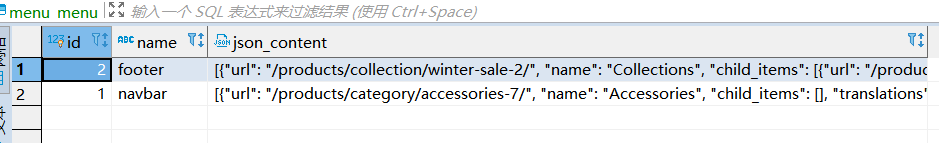


### 原因：

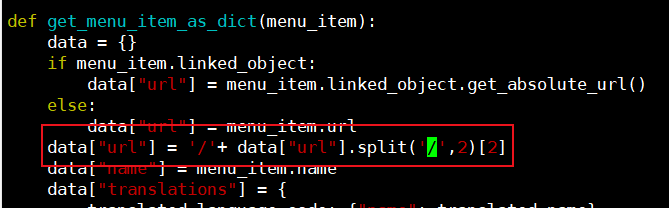


操作menu\_item时(就是上图中的表单对象)，在插入和更新时总会调用A(get\_menu\_item\_as\_dict())方法。

这个方法里面是获取数据的，项目这一小块就是用这个方法获取前端的链接地址，最后包装保存到数据库里面，在数据库里面的存储方式：

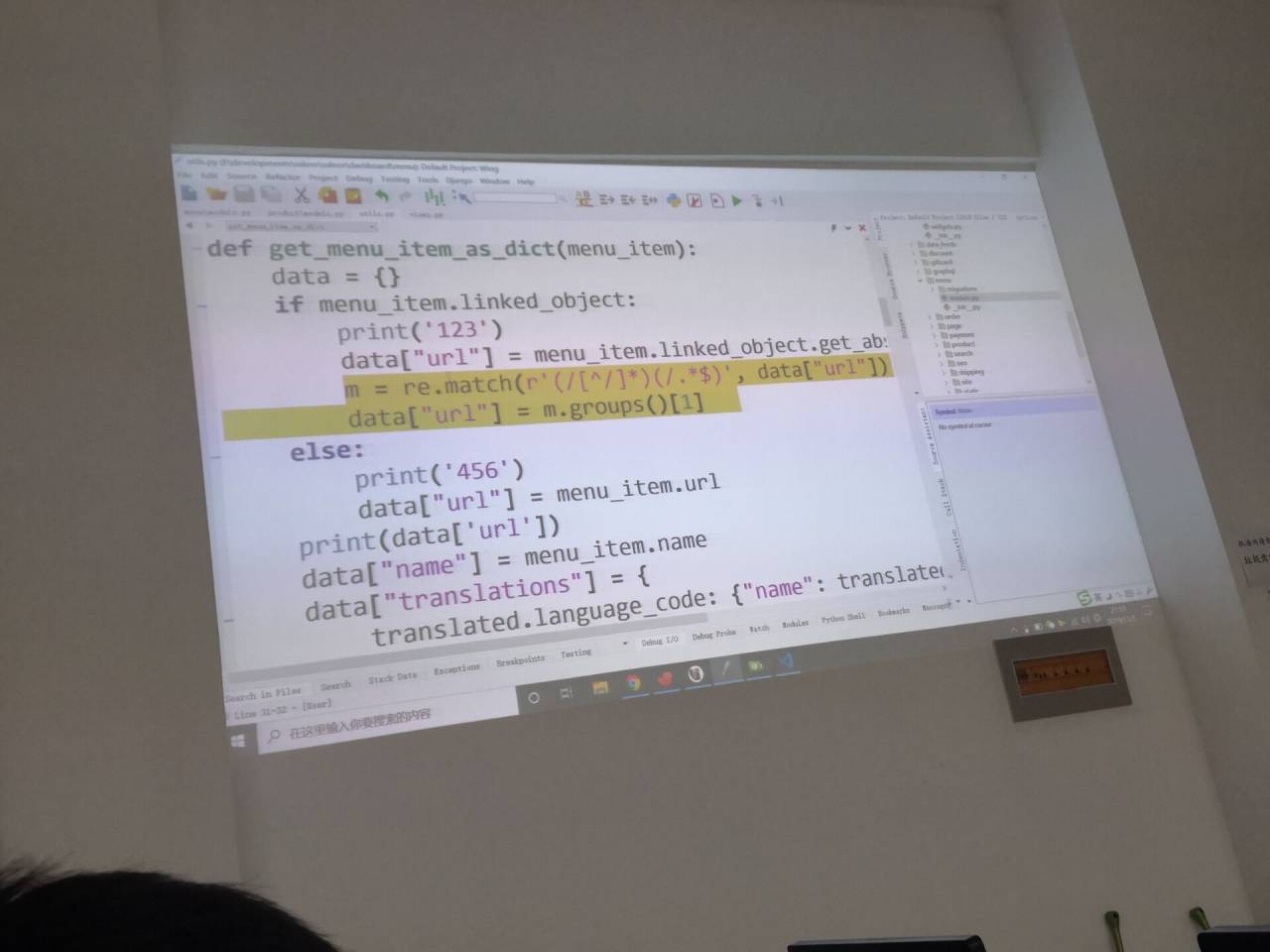


A方法就是获取数据字段json\_content中url的方法，其实它还有底层，就是get\_absolute\_url()方法，但是这个方法里面是django底层（Django的reverse方法）了，所以在A方法中修改如下：



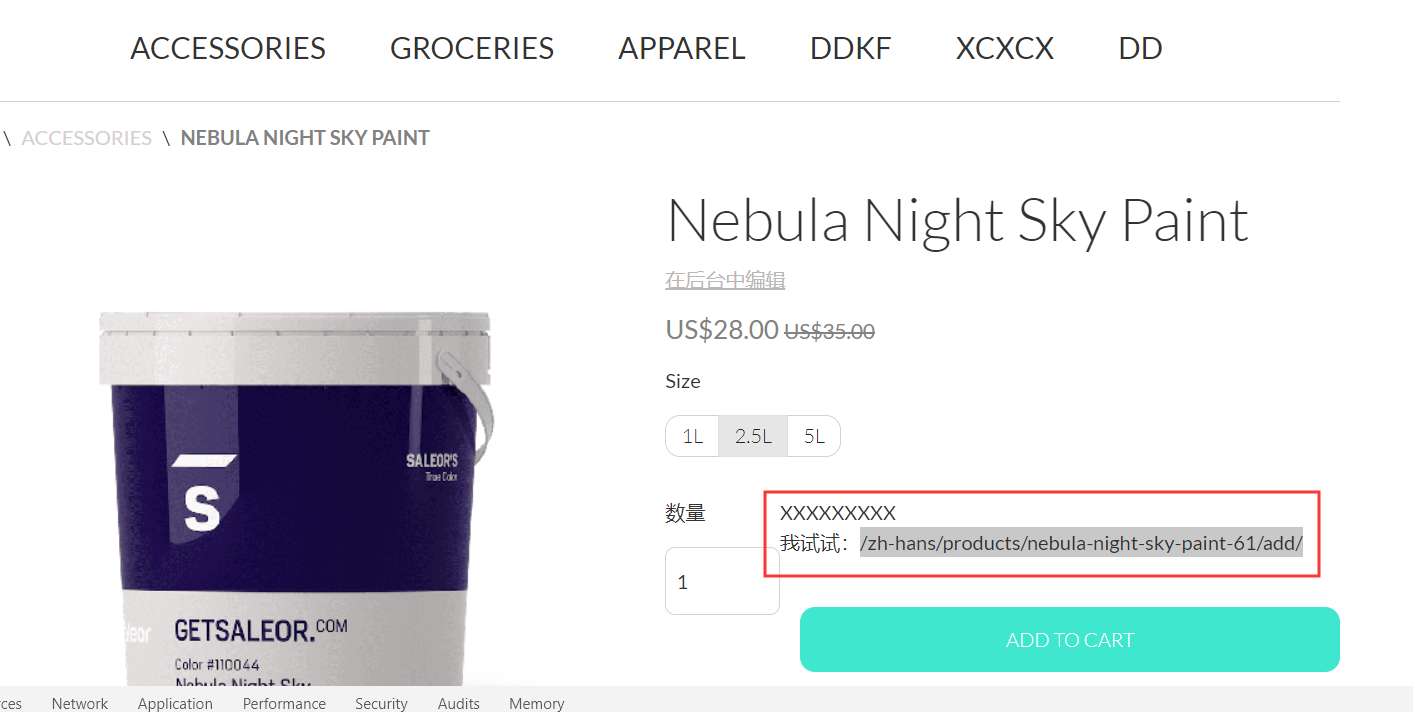
最后的原因是因为django的reverse方法对i8n支持有缺陷，获取的url带着/{languge\_code}/。

或者是这样改：下面这样改更符合理念。



# 2 单项上加字段

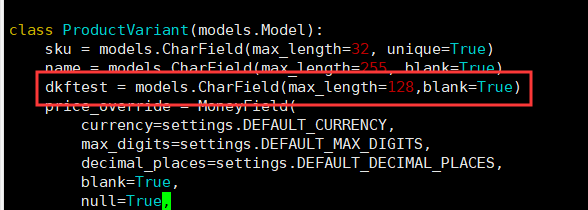
也就是增加一个变种字段。



这些数据是从数据库来的，所以如果不要死数据，就要从数据库上彻底更改，也就是说从model里面改。

## 第一步，从model里面改：

路径：家目录/saleor/saleor/product/models.py

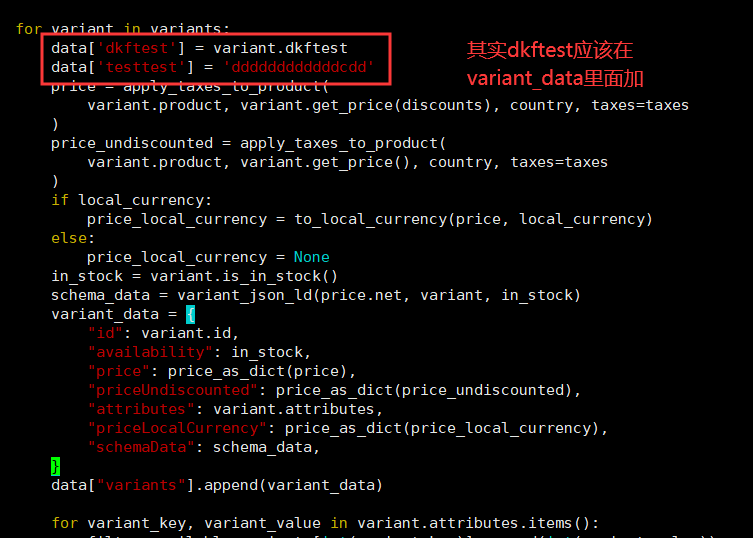


因为我是后加的一个属性，所以必须可为空，不然之前的记录上这个值是空，会报错。

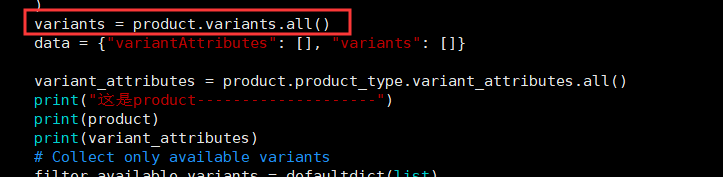
## 第二步，修改py文件：

路径：家目录/saleor/saleor/product/utils/variants\_picker.py

从数据库获取到数据表的数据，其中variants就是productVariant数据表的所有数据，也就是在上一步添加属性的地方（就是那个表）。



说明疑惑点：



productVariant数据库有外键和product表主键关联，所以这样获取是或者这个商品（product）的variant所有数据。

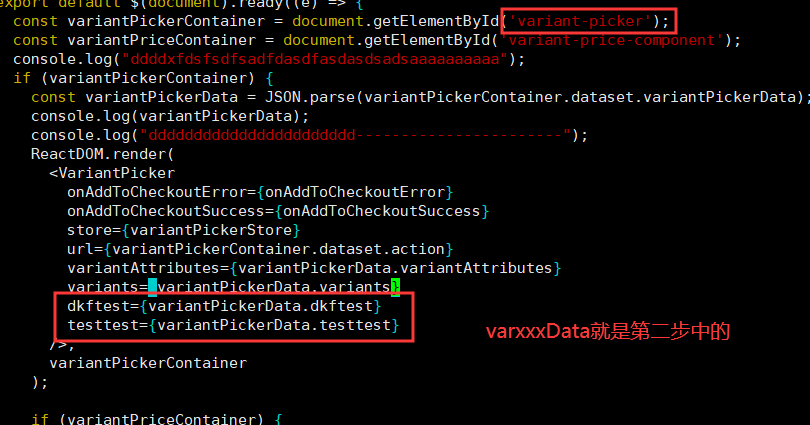
## 第三步：前后端的中间枢点

路径：家目录/saleor/templates/product/details.html

这是前端，第二步的文件那里，是view调用的方法，返回了data,然后到了这个页面，reactJs会获取红框这个标签，获得红框中的变量（对象）的值，然后进行渲染。

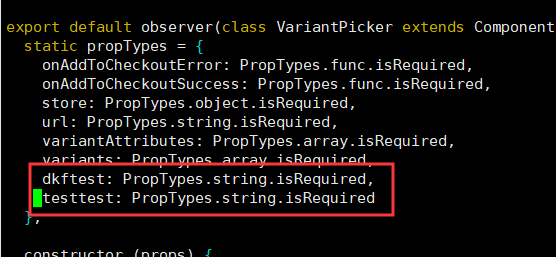
## 第四步，ReactJs渲染

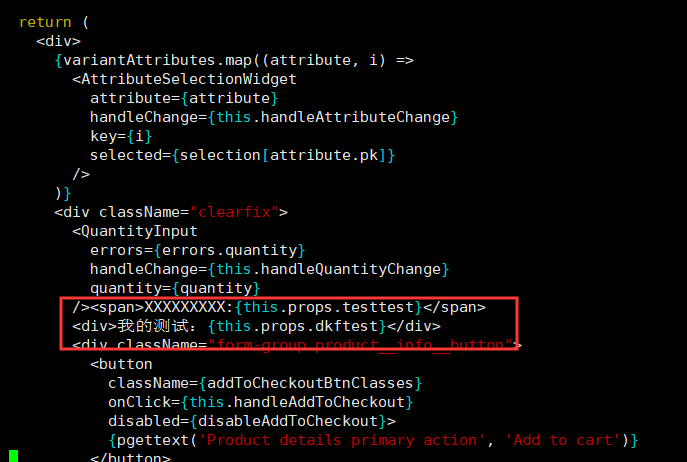
路径：家目录/saleor/saleor/static/js/components/variant-picker.js



第五步，最后渲染

路径：家目录/saleor/saleor/static/js/components/variantPicker/VariantPicker.js





## 第六步：重新编译前端

注意在数据库的productVariant数据表中写上添加字段的值，不然测试时这个字段还是空的，看不出来。然后再重新编译前端：

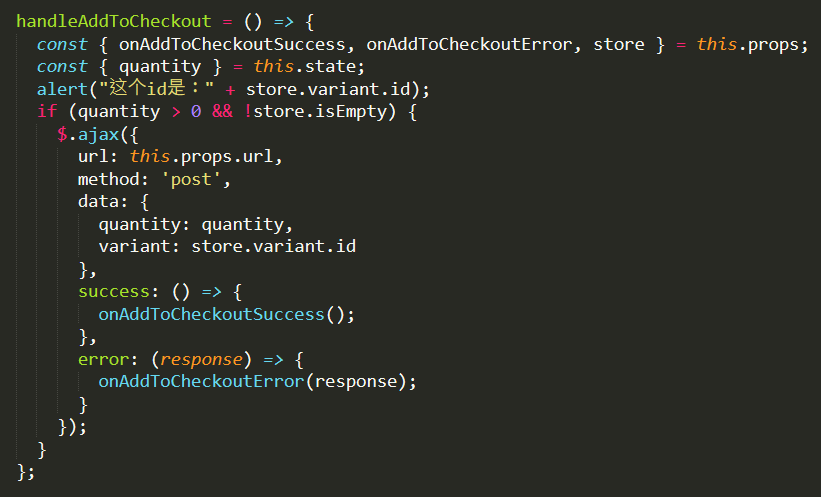
npm run build-assets && npm run build-emails

# 3 在购物车列表那里显示加的字段

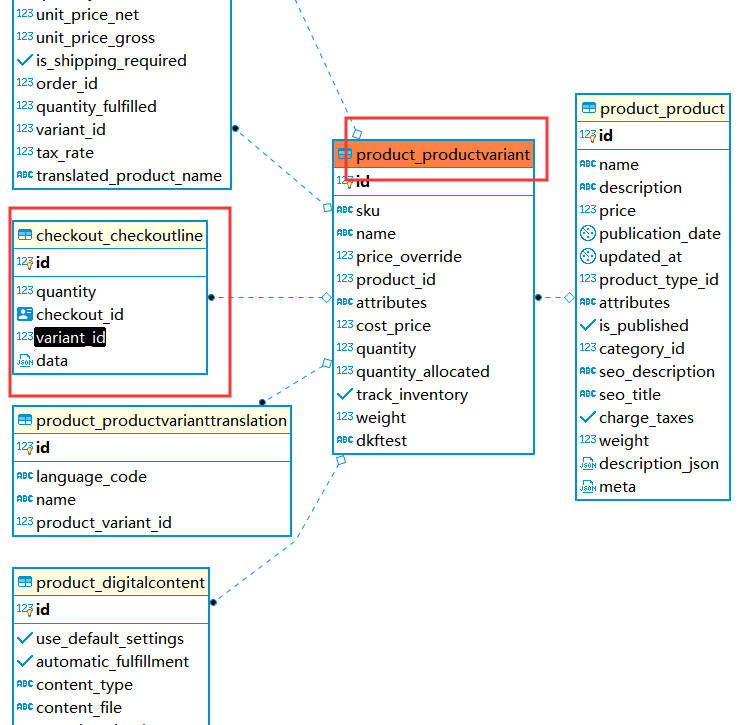
## 说明一些事情：

1. 点击【加入购物车】，会发送ajax请求，这个发送ajax请求的方法是React渲染的。其实就发送了购买的数量和变体（variant）记录的id。其实productrVariant数据表和checkout（购物车数据表进行了关联），如下图二所示：

路径：家目录/saleor/saleor/static/js/components/variantPicker/VariantPicker.js



图一

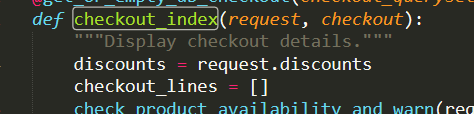


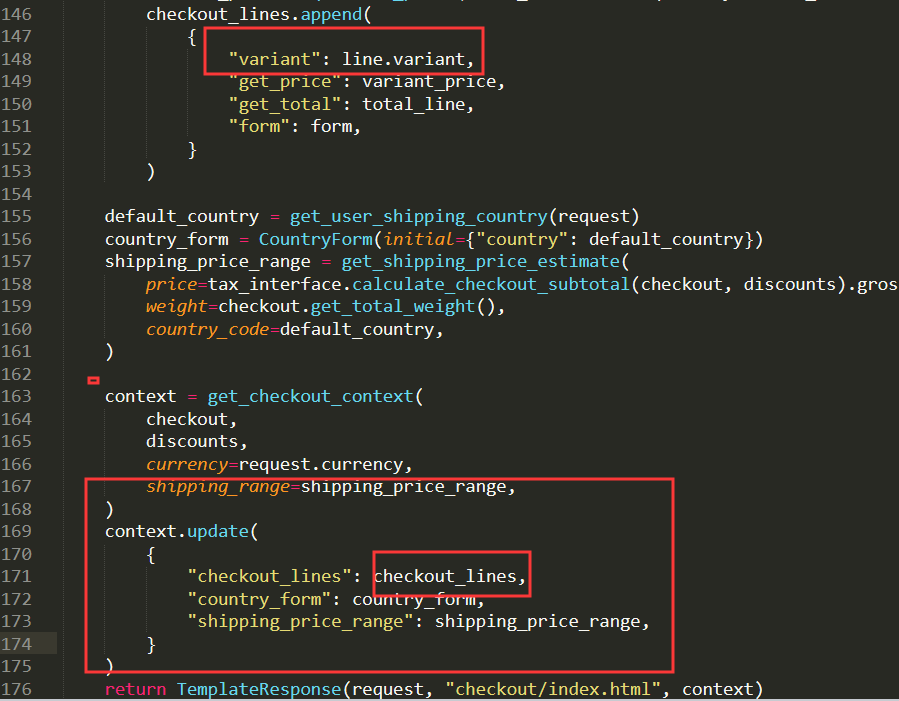
## 在购物车列表页面显示

### 后台：

前台的数据是如何获取到的。

路径：家目录/saloer/saloer/checkout/views/\_init\_.py

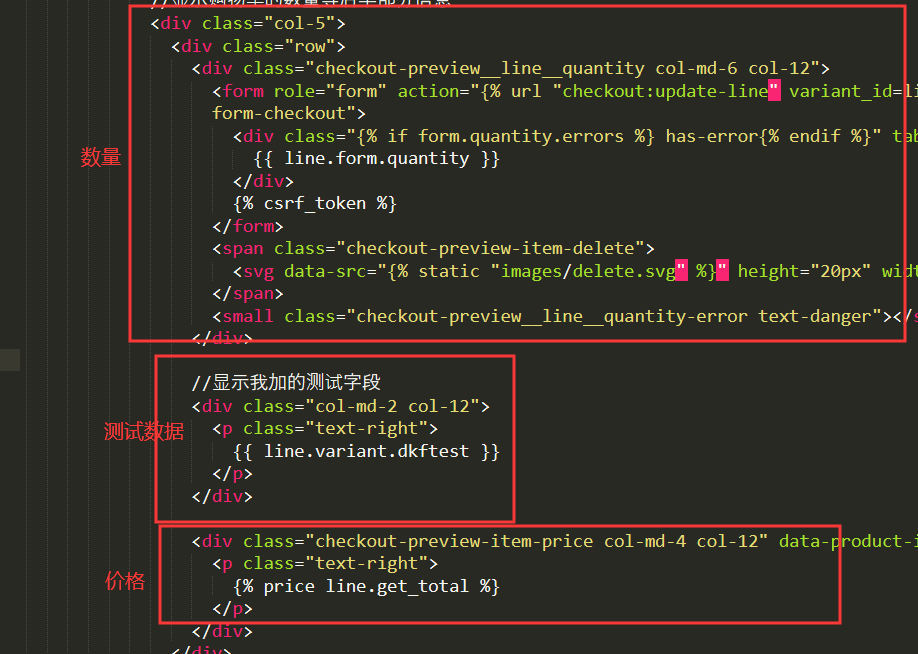




### 前台：

如何在前台渲染，就是用DJango语法循环显示出数据。

路径：家目录/saloer/templates/checkout/index.html



## 效果

