11.2 購物車

想要在網站上自己開店當老闆嗎？那麼購物車就是您網站必須具備的主要功能之一，本專題可以幫助您完成購物流程的規劃、製作及網路開店的夢想！

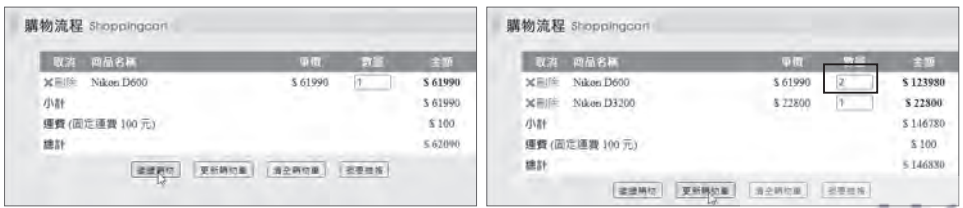
本專題將以顧客的購物流程製作為重點，至於商品庫存控管、客戶管理或是訂單處理等問題，則不在本專題討論範圍中。

11.2.1 使用購物車

將本章範例 cart 專案複製到複製到 <C:\example> 資料夾中，修改 <views.py> 中第 109-110 列為使用者的 Gmail 帳號及密碼，然後在命令提示字元視窗切換到 cart 專案，以「python manage.py runserver」啟動伺服器。

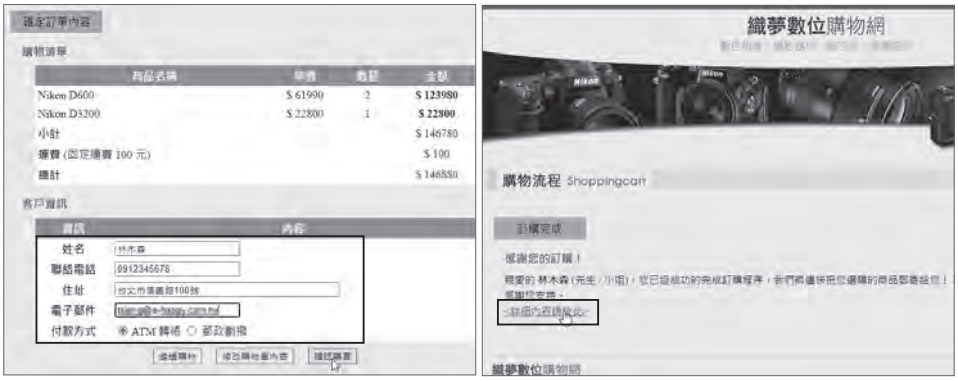
首頁會顯示所有商品照片、名稱及價格，此時尚未開始購物，會顯示「購物車是空的」訊息。點選商品照片或名稱會切換到該商品詳細頁面。在首頁或商品詳細頁面按 **加入購物車** 鈕可將商品加入購物車並切換到購物車頁面。

購物車頁面會顯示所有購買的商品及計算費用。按 **繼續購物** 鈕可回到首頁點選商品，若點選相同商品其數量會累加，也可修改 **數量** 欄位值後按 **更新購物車** 鈕，系統會重新計算費用。



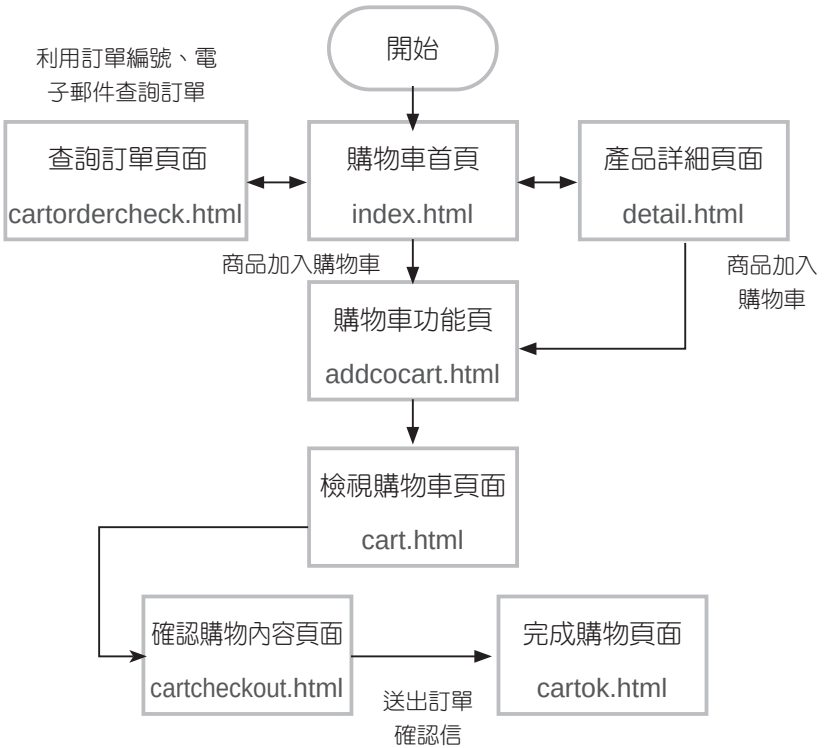
在購物車頁面按商品左方的 **刪除** 鈕會彈出確認對話方塊，按 **確定** 鈕就移除該項商品，按 **取消** 鈕就取消刪除功能後關閉對話方塊。

在購物車頁面按 **清空購物車** 鈕會將所有購買的商品移除，按 **我要結帳** 鈕就切換到 **確認訂單內容** 頁面：顯示購買商品清單，使用者輸入基本資料 (所有欄位都要輸入) 後按 **確認購買** 鈕，系統會寄電子郵件給購買者，使用者在 **訂購完成** 頁面按 **詳細內容請按此** 鈕就開啟 **訂單資料** 頁面。



訂單資料 頁面會顯示購買商品及購買者基本資料。在任何頁面按右方 **資訊中心** 的 **查詢訂單** 鈕開啟 **訂單查詢** 頁面，輸入收到電子郵件中的訂單編號及購買者電子郵件，如果兩者都正確才會切換到 **訂單資料** 頁面。

11.2.2 購物車流程圖



11.2.3 購物車資料庫結構

資料庫結構定義在 <models.py> 中，內含 ProductModel、OrdersModel 及 DetailModel 資料表：

cartapp\models.py

```
1 from django.db import models
2
3 class ProductModel(models.Model):
4   pname =  models.CharField(max_length=100, default='')
5   pprice = models.IntegerField(default=0)
6   pimages = models.CharField(max_length=100, default='')
7   pdescription = models.TextField(blank=True, default='')
8   def __str__(self):
9     return self.pname
10
```

```
11 class OrdersModel(models.Model):
12   subtotal = models.IntegerField(default=0)
13   shipping = models.IntegerField(default=0)
14   grandtotal = models.IntegerField(default=0)
15   customname =  models.CharField(max_length=100, default='')
16   customemail =  models.CharField(max_length=100, default='')
17   customaddress =  models.CharField(max_length=100, default='')
18   customphone =  models.CharField(max_length=100, default='')
19   paytype =  models.CharField(max_length=50, default='')
20   def __str__(self):
21     return self.customname
22
23 class DetailModel(models.Model):
24   dorder = models.ForeignKey('OrdersModel', on_delete=models.CASCADE)
25   pname = models.CharField(max_length=100, default='')
26   unitprice = models.IntegerField(default=0)
27   quantity = models.IntegerField(default=0)
28   dtotal = models.IntegerField(default=0)
29   def __str__(self):
30     return self.pname
```

ProductModel 資料表儲存線上購物的商品內容，所有欄位名稱都以「p」為前置字元：pname、pprice、pimages 及 pdescription 分別儲存商品名稱、價格、照片及說明。

OrdersModel 資料表儲存訂單資料：subtotal、shipping、grandtotal 儲存未加運費的購物金額、運費、加入運費的購物金額，customname、customemail、customaddress、customphone、paytype 儲存購買者姓名、電子郵件、地址、電話、付款方式。

DetailModel 資料表儲存詳細的訂單內容：dorder、pname、unitprice、quantity 及 dtotal 分別儲存訂單、商品名稱、單價、數量及總價。dorder 欄位為 OrdersModel 資料表的關聯欄位，「on_delete=models.CASCADE」表示移除 OrdersModel 中訂單時，該訂單中所有商品會自動移除。

11.2.4 Url 配置檔

cart\urls.py

```
略……
16 from django.contrib import admin
17 from django.urls import path
```

```
18 from cartapp import views
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('', views.index),
23     path('index/', views.index),
24     path('detail/<int:productid>/', views.detail),
25     path('addtocart/<str:ctype>/', views.addtocart),
26     path('addtocart/<str:ctype>/<int:productid>/', views.addtocart),
27     path('cart/', views.cart),
28     path('cartorder/', views.cartorder),
29     path('cartok/', views.cartok),
30     path('cartordercheck/', views.cartordercheck),
31 ]
```

程式說明

- 22-23　指定「127.0.0.1:8000」或「127.0.0.1:8000/index/」時執行 index 函式 （首頁）。
- 24　執行商品詳細頁面函式。
- 25-26　執行在購物車中加入、更新、刪除商品及清空購物車的函式。
- 27-30　28 列顯示購物車函式及顯示訂單函式，29 列為訂購完成函式，30 列為查詢訂單函式。

11.2.5 建立網頁基礎模版

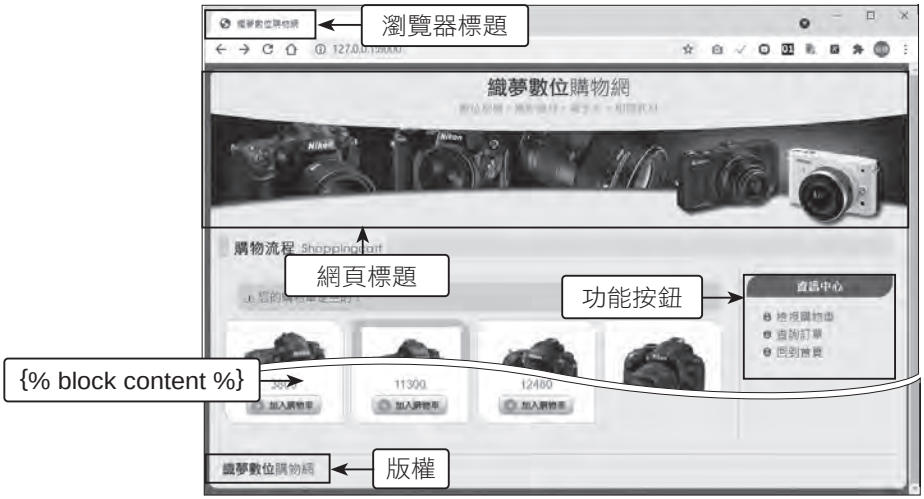
網站中所有網頁結構類似，為這些網頁建立基礎模版檔 <base.html>：包括瀏覽器標題、網頁標題、右方 **資訊中心** 功能按鈕及下方的版權部分。

templates\base.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5   <title> 織夢數位購物網 </title>
6   {% load static %}
7   <link href="{% static "css/style.css" %}" rel="stylesheet"
          type="text/css" />
8     {% block script %}{% endblock %}
9 </head>
10 <body>
```

```
11   <div id="warp">
12     <div id="header">
13       <div class="logo"></div>
14     </div>
15     <div id="content">
16       <img src="{% static "images/cartTitle2.png" %}" width="920"
                 height="29" />
17       <div id="sidediv">
18         <div class="sidemenu"> 資訊中心 </div>
19         <ul>
20           <li><a href="/cart/"> 檢視購物車 </a></li>
21           <li><a href="/cartordercheck/"> 查詢訂單 </a></li>
22           <li><a href="/index/"> 回到首頁 </a></li>
23         </ul>
24       </div>
25       {% block content %}{% endblock %}
26       <div style="clear:both"></div>
27     </div>
28     <div id="footer"></div>
29   </div>
30 </body>
31 </html>
```

「{% block content %}」設定網頁內容。



11.2.6 首頁處理函式及模版

使用者以「http://127.0.0.1:8000」開啟本專題就會執行 <views.py> 中的 index 函式載入 <index.html> 網頁，會顯示所有商品。

cartapp\views.py

```
 1 from django.shortcuts import render, redirect
 2 from cartapp import models
 3 from smtplib import SMTP, SMTPAuthenticationError, SMTPException
 4 from email.mime.text import MIMEText
 5
 6 message = ''
 7 cartlist = []  # 購買商品串列
 8 customname = ''  # 購買者姓名
 9 customphone = ''  # 購買者電話
10 customaddress = ''  # 購買者地址
11 customemail = ''  # 購買者電子郵件
12
13 def index(request):
14   global cartlist
15   if 'cartlist' in request.session: # 若 session 中存在 cartlist 就讀出
16     cartlist = request.session['cartlist']
17   else:  # 重新購物
18     cartlist = []
19   cartnum = len(cartlist)  # 購買商品筆數
20   productall = models.ProductModel.objects.all() # 取得資料庫所有商品
21   return render(request, "index.html", locals())
略……
```

程式說明

- 3-4　本專題使用 Smtplib 模組寄送電子郵件，故需匯入這些模組。
- 6-11　訊息 (message)、購物車商品串列 (cartlist) 及購買者資訊 (customname 等)，有多個頁面會使用這些資料，故設為全域變數。
- 15-18　為了讓使用者已選取的購物商品在瀏覽器關閉之後仍能存在一段時間，因此本專題將購物車商品存於 session 中。第 15-16 列檢查 session 購物車中是否有商品存在，若有就讀取資料讓使用者繼續購物，如果 session 沒有購物車商品，表示使用者重新購物，在 17-18 列清空購物車串列。
- 19　取得購物車商品數量，傳送給 <index.html> 判斷購物車商品的數量。
- 20　由資料庫取得所有商品資料，傳送給 <index.html> 顯示商品資訊。

templates\index.html

```
1 {% extends "base.html" %}
2 {% load static %}
3
4 {% block content %}
5   <div id="maindiv">
6     {% if cartnum == 0 %}
7       <div class="message"> 您的購物車是空的！ </div>
8     {% endif %}
9     {% for product in productall %}
10      <div class="block_product pro_photo">
11        <div class="pro_photo">
12          <a href="/detail/{{product.id}}"><img src="/static/
                    images/{{product.pimages}}" alt="產品詳細資料"
                    width="135" height="135" border="0" /></a>
13          <div class="pro_name"><a href="/detail/{{product.id}}">
                    {{product.pname}}</a></div>
14          <div class="pro_price">{{product.pprice}}</div>
15          <div class="pro_btn">
16            <a href="/addtocart/add/{{product.id}}/"><img src="{% static
                      "images/icon_cart.png" %}" height="27"
                      border="0" /></a>
17          </div>
18        </div>
19      </div>
20    {% endfor %}
21  </div>
22 {% endblock %}
```

程式說明

- 1　繼承 <base.html> 基礎模版。
- 2　如果網頁中有使用 <static> 資料夾的靜態檔案，必須以「{% load staticfiles %} 」載入靜態檔。
- 4-22　為網頁內容。
- 6-8　判斷購物車是否有購買的商品，若沒有就顯示「購物車是空的」訊息。
- 9-20　逐筆顯示商品：12 列顯示商品照片，13 列顯示商品名稱，兩者都設定連結開啟商品詳細頁面 (/detail/{{product.id}})，14 列顯示商品價格，15-17 列建立 **加入購物車** 按鈕。

11.2.7 商品詳細頁面處理函式及模版

使用者在首頁按圖片或商品名稱就會執行 <views.py> 中的 detail 函式載入 <detail.html> 網頁顯示商品的詳細資訊。

cartapp\views.py

```
略……
23 def detail(request, productid=None):  # 商品詳細頁面
24   product = models.ProductModel.objects.get(id=productid) # 取得商品
25   return render(request, "detail.html", locals())
略……
```

第 23 列參數 productid 為商品編號，24 列根據 productid 取得該商品資訊傳送給 <detail.html> 網頁。

templates\detail.html

```
 1 {% extends "base.html" %}
 2 {% load static %}
 3
 4 {% block content %}
 5   <div id="maindiv">
 6     <table width="100%" border="0" align="center" cellpadding="4"
             cellspacing="0">
 7       <tr valign="top">
 8         <td width="200" align="center"><p><img src="/static/
                  images/{{product.pimages}}" />
 9           <p class="sprice"> 特價 <br />${{product.pprice}}</p></td>
10         <td><p class="subject"> 產品名稱 </p>
11           <p class="desc">{{product.pname}} </p>
12           <p class="subject"> 產品介紹 </p>
13           <p class="desc">{{product.pdescription}}</p></td>
14       </tr>
15     </table>
16     <p align="center">
17       <a href="/addtocart/add/{{product.id}}/"><img src="{% static
                "images/icon_cart.png" %}" width="102" height="27"
                border="0" /></a>
18       <a href="javascript:;" onclick="window.history.back();">
                <img src="{% static "images/icon_back.png" %}" width="102"
                height="27" border="0" /></a>
19     </p>
```

```
20    </div>
21 {% endblock %}
```

程式說明

- 8　　顯示商品照片。
- 9-13　　顯示商品價格、名稱及說明。
- 17　　建立 **加入購物車** 按鈕。
- 18　　建立 **回到上一頁** 按鈕。

11.2.8 顯示購物車頁面處理函式及模版

使用者在任何頁面按右方 **檢視購物車** 或在首頁、商品詳細頁面按 **加入購物車** 鈕就會執行 <views.py> 中的 cart 函式載入 <cart.html> 網頁，顯示購買商品的詳細資訊並可修改購買數量。

cartapp\views.py

```
略……
27 def cart(request):  # 顯示購物車
28    global cartlist
29    cartlist1 = cartlist  # 以區域變數傳給模版
30    total = 0
31    for unit in cartlist:  # 計算商品總金額
32       total += int(unit[3])
33    grandtotal = total + 100  # 加入運費總額
34    return render(request, "cart.html", locals())
略……
```

第 28 列取得全域變數 cartlist，29 列以區域變數 cartlist1 傳送購物車資料給 <cart.html>。30-32 列計算購買商品價錢總額：每一筆購買商品資料的第 4 項為該商品總價 (unit[3])，逐項將商品總價相加就是購買商品總額。33 列加上運費。

cart 函式僅計算總價傳送給 <cart.html> 顯示，購物車中商品的增添、修改及刪除等功能則由下一小節的 addtocart 函式處理。

templates\cart.html

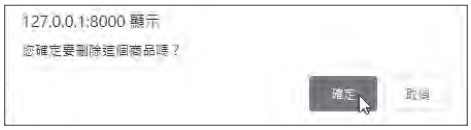
```
1 {% extends "base.html" %}
2 {% load static %}
3
4 {% block script %}
```

```
 5   <script type="text/javascript">
 6     function confirmLink(message) { //v1.0
 7     if(message == "") message = "確定";
 8       document.returnValue = confirm(message);
 9     }
10   </script>
11 {% endblock %}
12
13 {% block content %}
14   <div id="maindiv">
15     <form action="/addtocart/update/" method="post" name="form1" id="form1">
16       {% csrf_token %}
17       <table width="90%" border="0" align="center" cellpadding="4"
                cellspacing="0">
18         <tr>
19           <th width="60" align="center"><strong> 取消 </strong></th>
20           <th align="left"><strong> 商品名稱 </strong></th>
21           <th width="80" align="center"><strong> 單價 </strong></th>
22           <th width="80" align="center"><strong> 數量 </strong></th>
23           <th width="100" align="center"><strong> 金額 </strong></th>
24         </tr>
25         {% for unit in cartlist1 %}
26         <tr>
27           <td bgcolor="#FFFFFF"><a href="/addtocart/remove/
                  {{forloop.counter0}}" class="delcart" onClick=
                  "confirmLink(' 您確定要刪除這個商品嗎？ ');
                  return document.returnValue"> 刪除 </a></td>
28           <td align="left" bgcolor="#FFFFFF">{{unit.0}}</td>
29           <td width="80" align="center" bgcolor="#FFFFFF">$ {{unit.1}}</td>
30           <td width="80" align="center" bgcolor="#FFFFFF">
                  <input name="qty{{forloop.counter0}}" type="text"
                  id="qty{{forloop.counter0}}" value="{{unit.2}}"
                  size="2" /></td>
31           <td width="100" align="center" bgcolor="#FFFFFF">
                  <strong>$ {{unit.3}}</strong></td>
32         </tr>
33         {% endfor %}
34         <tr>
35           <td colspan="4" align="left" bgcolor="#FFFFFF"
                    class="upline"><strong> 小計 </strong></td>
36           <td align="center" bgcolor="#FFFFFF" class="upline">
                    $ {{total}}</td>
```

```
37          </tr>
38          <tr>
39            <td colspan="4" align="left" bgcolor="#FFFFFF" class="upline">
                      <strong> 運費 </strong> ( 固定運費 100 元 ) </td>
40            <td width="100" align="center" bgcolor="#FFFFFF"
                      class="upline">$ 100</td>
41          </tr>
42          <tr>
43            <td colspan="4" align="left" bgcolor="#FFFFFF">
                      <strong> 總計 </strong></td>
44            <td align="center" bgcolor="#FFFFFF"><strong><font color=
                      "#FF0000">$ {{grandtotal}}</font></strong></td>
45          </tr>
46        </table>
47        <table border="0" align="center" cellpadding="10" cellspacing="0">
48          <tr>
49            <td><input type="button" name="Continue" value=" 繼續購物 "
                      onclick="window.location='/index/'" /></td>
50            <td><input name="Submit" type="submit" id="Submit"
                      value=" 更新購物車 "></td>
51            <td><input name="Empty" type="button" id="Empty"
                      onclick="window.location='/addtocart/empty/'"
                      value=" 清空購物車 " /></td>
52            <td><input name="Order" type="button" id="Order" onclick=
                      "window.location='/cartorder/'" value="我要結帳" /></td>
53          </tr>
54        </table>
55      </form>
56    </div>
57 {% endblock %}
```

程式說明

- 4-11　建立確認刪除彈出式對話方塊的 JavaScript 函式：使用者在對話方塊中按 **確定** 鈕就傳回 True，按 **取消** 鈕就傳回 False。然後在 27 列使用此函式：若傳回 True 就以「/addtocart/remove/{{forloop.counter0}}」刪除購物車中的商品，若傳回 False 就不刪除商品。



- 16　因為本網頁需以 POST 方式送出表單，所以要加入 CSRF 驗證。

- 18-24　建立表格標題。
- 25-33　逐筆顯示購物車中商品資訊:27列建立 **刪除** 按鈕,28 列顯示商品名稱,29 列顯示商品價格,30 列顯示商品數量,為了讓使用者可以修改商品數量,此欄位以文字輸入框呈現,31 列顯示商品總價。
- 34-37　顯示購物車所有商品總金額。
- 38-41　顯示運費,本專題的運費固定為 100 元。
- 42-45　加上運費的總金額。
- 49-52　建立 4 個功能按鈕。

11.2.9 購物車處理函式

購物車處理函式 (addtocart) 負責購物車中商品的增添、修改及刪除等功能，本函式沒有顯示頁面，處理完相關功能後會切換到相關頁面。

首先是使用者在首頁或商品詳細頁面按 **加入購物車** 鈕的處理程式，功能是將使用者選取的商品加入購物車。

cartapp\views.py

```
略……
def addtocart(request, ctype=None, productid=None):
  global cartlist
  if ctype == 'add':  # 加入購物車
    product = models.ProductModel.objects.get(id=productid)
    flag = True  # 設檢查旗標為 True
    for unit in cartlist:  # 逐筆檢查商品是否已存在
      if product.pname == unit[0]:  # 商品已存在
        unit[2] = str(int(unit[2])+ 1)  # 數量加 1
        unit[3] = str(int(unit[3]) + product.pprice)  # 計算價錢
        flag = False  # 設檢查旗標為 False
        break
    if flag:  # 商品不存在
      temlist = []  # 暫時串列
      temlist.append(product.pname)  # 將商品資料加入暫時串列
      temlist.append(str(product.pprice))  # 商品價格
      temlist.append('1')  # 先暫訂數量為 1
      temlist.append(str(product.pprice))  # 總價
      cartlist.append(temlist)  # 將暫時串列加入購物車
    request.session['cartlist'] = cartlist  # 購物車寫入 session
    return redirect('/cart/')
略……
```

程式說明

- 36-39　參數 ctype 值代表執行不同功能：38 列「ctype=='add'」表示執行加入購物車功能，productid 為使用者選取商品的編號，39 列根據 productid 取得選取商品的資料。
- 40　「flag=True」表示選取的商品在購物車中不存在。購物車中是否已有選取商品的處理方式不同：若購物車中沒有選取商品就要在購物車中新增一筆商品資料，若已有選取商品則將該商品的數量加 1 即可。
- 41-46　逐一檢查購物車中是否已有選取商品：42-44 列為選取商品已存在就將數量加 1 並計算新的商品總價，45 列設 flag 為 False，表示商品已存在，46 列中斷迴圈，不必再檢查。
- 47-53　為購物車中沒有選取商品的處理程式：47 列 flag 為 True 表示購物車中沒有選取商品，48-52 列以一個暫時串列儲存選取商品的資料，53 列將暫時串列加入購物車。
- 54-55　將購物車寫入 session，再切換到顯示購物車頁面。

接著是使用者在顯示購物車頁面按 **更新購物車** 鈕的處理程式。

cartapp\views.py

```
略……
56 elif ctype == 'update':  # 更新購物車
57     n = 0
58     for unit in cartlist:
59         unit[2] = request.POST.get('qty' + str(n), '1')  # 取得數量
60         unit[3] = str(int(unit[1]) * int(unit[2]))  # 取得總價
61         n += 1
62     request.session['cartlist'] = cartlist
63     return redirect('/cart/')
略……
```

程式說明

- 56　參數「ctype=='update'」表示執行更新購物車功能。
- 57　變數 n 為計數器，可以在 59 列取得數量輸入欄位的值。
- 58-61　逐項處理商品資料：59 列取得數量輸入欄位的值，60 列重新計算商品總價。

再來是使用者在顯示購物車頁面按 **清空購物車** 鈕的處理程式。

cartapp\views.py

```
略……
64 elif ctype == 'empty':  # 清空購物車
65     cartlist = []  # 設購物車為空串列
66     request.session['cartlist'] = cartlist
67     return redirect('/index/')
略……
```

只要將購物車串列 (cartlist) 設為空串列再寫入 session 即可。

最後是使用者在顯示購物車頁面按商品名稱左方 **刪除** 鈕的處理程式。

cartapp\views.py

```
略……
68 elif ctype == 'remove':  # 刪除購物車中商品
69     del cartlist[int(productid)]  # 從購物車串列中移除商品
70     request.session['cartlist'] = cartlist
71     return redirect('/cart/')
略……
```

程式說明

- 69　　刪除選取商品後將購物車串列 (`cartlist`) 寫入 `session`。

11.2.10 訂單頁面處理函式及模版

使用者在顯示購物車頁面點選 **我要結帳** 鈕就會執行 <views.py> 中的 cartorder 函式載入 <cartorder.html> 網頁讓使用者填寫購買者資訊。

cartapp\views.py

```
略……
73 def cartorder(request):  # 按我要結帳鈕
```

```
74    global cartlist, message, customname, customphone, customaddress,
            customemail
75    cartlist1 = cartlist
76    total = 0
77    for unit in cartlist:  # 計算商品總金額
78       total += int(unit[3])
79    grandtotal = total + 100
80    customname1 = customname  ## 以區域變數傳給模版
81    customphone1 = customphone
82    customaddress1 = customaddress
83    customemail1 = customemail
84    message1 = message
85    return render(request, "cartorder.html", locals())
略……
```

程式說明

- 74　取得全域變數購物車串列及購買者資訊。
- 75　將區域變數傳送給 <cartorder.html> 顯示。
- 76-78　逐一加總各商品總價得到購買總金額。
- 79　加上運費。
- 80-84　將區域變數傳送給 <cartorder.html> 顯示。

templates\cartorder.html

```
1 {% extends "base.html" %}
2 {% load static %}
3
4 {% block content %}
5   <div id="maindiv">
6     <form action="/cartok/" method="POST" name="form1" id="form1">
7       {% csrf_token %}
8       <p class="title"> 確定訂單內容 </p>
9       <p class="subject"> 購物清單 </p>
略…… (10-37 列顯示購買商品及總價 )
38      <p class="subject"> 客戶資訊 </p>
39      <table width="90%" border="0" align="center" cellpadding="4"
                cellspacing="0">
40        <tr>
41          <th width="100" align="center"> 資訊 </th>
42          <th> 內容 </th>
43        </tr>
44        <tr>
```