Integrating ProductCart v3 with Your Web site

Table of Contents

Overview

Many e-commerce applications limit your ability to customize the look & feel of your online store, therefore preventing you from blending store pages with the rest of your Web site. ProductCart doesn't. ProductCart allows you to build an online store that will merge seamlessly with your Web site.

In other words, you can use virtually any graphical interface that you can think of. If you already have a Web site, it will be a snap for you to make all shopping cart pages blend with the rest of the site. If you don't have a Web site yet, you can be as creative as you'd like. ProductCart does not limit you in any way. By taking a look at a few of them (see http://www.greatonlinestores.com), you will get a good idea of how many other users of the application have integrated ProductCart with their Web site designs.

By reading the rest of this document, you will acquire all the information you need to take advantage of a number of ProductCart features that allow you seamlessly merge the shopping cart with the rest of your Web site. A basic understanding of HTML is recommended.

Header.asp & Footer.asp

Every storefront file that is part of ProductCart "includes" two other files that define the way it is presented to store visitors. The files are called *header.asp* and *footer.asp*. Because the same files are included with every page generated by ProductCart in your storefront, you only need to worry about creating those two files to apply a consistent graphical interface to your entire storefront.

Generating the header and footer is a simple task. You only need to follow the steps mentioned below and you will be applying a custom graphical interface to your ProductCart store in no time.

In many areas of this document we will refer to the "default" header.asp and footer.asp files that are included in ProductCart. When you first install ProductCart, these are the files that make up the graphical interface of the storefront. If you no longer have these files (e.g. you overwrote them while generating your own) and want to download them again, you can do so by using this link:

http://www.earlyimpact.com/productcart/support/ProductCart_v3_header_footer.zip

Let's go back to the steps that you need to take to create your own *header.asp* and *footer.asp*.

First Step: create a template.

A template is typically a Web page that contains the main graphic and navigation elements that make up a Web site's interface. Creating a template is a very simple thing to do.

- First, launch your favorite HTML editor (e.g. Microsoft® FrontPage, Macromedia® DreamWeaver, Adobe® GoLive, etc.) and open any page that is part of your current Web site (or create a new page).
- Next, remove all the content that is specific to that page only. For example, your "about.html" page will likely contain some text that describes your business. Remove that text, but keep navigation and other graphic elements that are repeated on other pages.
- Save the file with a different name (e.g. template.html) and you've gotten yourself a template. By following the steps described over the next few pages, you will be able to quickly "apply" this template to your ProductCart-powered Web store.

Note to Macromedia Dreamweaver® Users: When we say 'template' in this document, we are not referring to a Dreamweaver template (*.dwt files). However, if you created a Dreamweaver template for your Web site, you can use it here too. Just follow these simple steps:

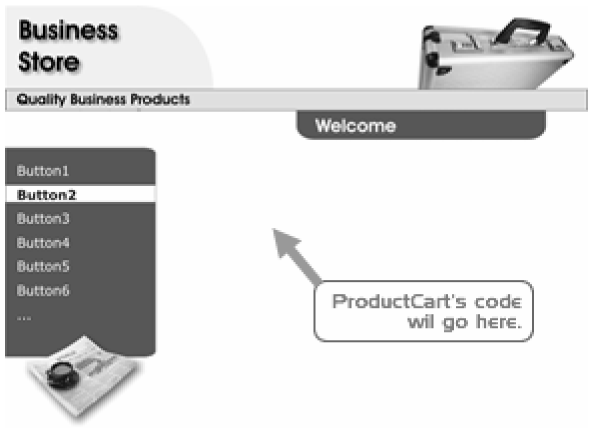
(1) Create a new page based on the Dreamweaver template that you created.

(2) Save the page to the *productcart/pc* folder in your Web site. This way Dreamweaver will automatically recalculate all the links for you.

(3) Detach the page from the template by selecting "Modify/Templates/Detach from Template". This page will no longer be controlled by your Dreamweaver template. However this is a necessary step to remove the extra code that Dreamweaver adds to the file's source code.

(4) Save the file and continue with the steps outlined below.

For example, the blank Web page shown below features a few graphic elements at the top, and a navigation menu on the left side. The page content would typically go in the white area in the center of the window, where we placed the "ProductCart's code will go here" message.



When you create this page, setup the links and image locations considering that the *header.asp* and *footer.asp* files that you will create from it will ultimately be located in the *pc* subfolder of the *productcart* folder. Do this to ensure that you will not end up with broken image links on your store interface. You have two options:

- Relative image paths. Assuming that your Web site images are stored in a folder called *images* located in the root directory of your Web site, and assuming that the *productcart* folder is also located in the root, the path to those images should be "../../images/myImage.gif".
- Absolute image paths. You can use an absolute path from the root folder. Assuming that you have positioned the *productcart* folder in the root of the Web site, the image links would be "/images/myImage.gif".

A third option is to use absolute URLs (e.g. http://www.myStore.com/images/myImage.gif), but this is not recommended as it can trigger a security alert if you use an SSL certificate on your store. When customers begin the secure checkout, they would be shown an alert as the browser would notice that one or more elements on the page (your images) are being loaded on the HTTP protocol instead of the HTTPS protocol.

Second Step: add the ASP code.

Now that you have created a "template" and edited the image links so that they will not be broken when *header.asp* and *footer.asp* are moved into the *pc* subfolder, let's focus on the next step, which consists of adding some lines of ASP code to your HTML file so that it properly communicates with ProductCart.

Your HTML template will likely begin with something like the following (this is the code created by Dreamweaver when you generate a new HTML file and is common to a lot of Web pages). Your page might have slightly different code (e.g. it might not include a "DocType" at the top, and/or include additional Meta Tags).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
   <title>My E-Commerce Store</title>
 </head>
```

This is called the "head" section of an HTML page. It includes a definition of what the document is (the first line of the code), the opening HTML and HEAD tags, a special Meta Tag that further defines the type of document, the "Title" Meta Tag, and the closing HEAD tag.

You will need to **replace** these lines of code with the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--#include file="include-metatags.asp"-->
<html>
 <head>
  <%if pcv_PageName<>"" then%>
   <title><%=pcv_PageName%></title>
  <%end if%>
  <%GenerateMetaTags()%>
  <%Response.Buffer=True%>
  <%
   Set conlayout=Server.CreateObject("ADODB.Connection")
   conlayout.Open scDSN
   Set RSlayout = conlayout.Execute("Select * From layout Where layout.ID=2")
   Set rsIconObj = conlayout.Execute("Select * From icons WHERE id=1")
  %>
  <link type="text/css" rel="stylesheet" href="pcStorefront.css" />
 </head>
```

The bottom of your template will typically end with the following code:

```
 </body>
</html>
```

You will need to **replace** these two lines with the following:

```
 <%
 conlayout.Close
 Set conlayout=nothing
 Set RSlayout = nothing
 Set rsIconObj = nothing
 %>
 </body>
</html>
```

<u>**Note**</u>: If there are extra lines or other characters in the code, ProductCart may not function properly. Sometimes pasting from a MS Word or PDF document directly into

your HTML editor can create problems. Instead of copying and pasting the code from this document, we recommend that open in your HTML editor the default *header.asp* and *footer.asp* files located in the *pc* folder, and copy the code from there. Or you can find the same code in this ZIP file.

Later in this document you will find a detailed explanation of what this code does. For now, please remember that:

- You need to replace your existing Title, Description, and Keywords Meta Tags with the code above, as ProductCart will automatically create Meta Tags for your product and category pages for you.
- If you create your header.asp and footer.asp files, load a ProductCart page, and receive an error, then probably the culprit is extra characters that were added to the code when you copied it and pasted it. Make sure that you are using the correct code. You can find it in this ZIP file.

Third Step: create "header.asp" and "footer.asp".

Now that you have added the required code to your template, you are ready to create *header.asp* and *footer.asp.* To do so, place your mouse cursor at the beginning of the section where the page content should appear. If you are using an HTML editor such as Macromedia® Dreamweaver or Microsoft® FrontPage, you can do this in the design view by positioning the cursor at the center of the table cell that is going to contain the shopping cart content.

Now switch to the HTML view to display the code. Copy all the code from that point to the beginning of the page, paste it into a new Notepad file, and save it as "**header.asp**". Go back to the template file and now copy all the code from that point to the end of the document, paste it into a new Notepad file, and save it as "**footer.asp**".

Your copy of ProductCart ships with default *header.asp* and *footer.asp* files. Use these files as a point of reference if the paragraphs above were not clear to you.

An Example

Let's look at a basic example of how this process works. The sample page that we are using for this example is an extremely simply HTML page that organizes its content in a table cell. The code for page, our "template page", would look like this:

```
<html>
  <head>
    <title>My Template</title>
  </head>
<body>
 <table>
  <tr>
   <td>
     This is my template
     </td>
    </tr>
   </table>
  </body>
</html>
```

In this basic example, the code for *header.asp* would look like this:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--#include file="include-metatags.asp"-->
<html>
 <head>
  <%if pcv_PageName<>"" then%>
   <title><%=pcv_PageName%></title>
  <%end if%>
  <%GenerateMetaTags()%>
  <%Response.Buffer=True%>
  <%
   Set conlayout=Server.CreateObject("ADODB.Connection")
   conlayout.Open scDSN
   Set RSlayout = conlayout.Execute("Select * From layout Where layout.ID=2")
   Set rsIconObj = conlayout.Execute("Select * From icons WHERE id=1")
  %>
  <link type="text/css" rel="stylesheet" href="pcStorefront.css" />
 </head>
 <body>
  <table>
   <tr>
    <td>
```

Notice how the code in *header.asp* stops with the opening tag for the table cell that in our template page contained the copy “This is my template”. We removed that text because we don’t want it to show on the store.

The code for *footer.asp* starts with the closing tag for the same table cell:

```
   </td>
  </tr>
 </table>
<%
 conlayout.Close
 Set conlayout=nothing
 Set RSlayout = nothing
 Set rsIconObj = nothing
 %>
 </body>
</html>
```

Fourth Step: upload the files.

Upload both files to the *productcart/pc/* folder on your Web server. You will see that the default version of those files is already located in that folder. Rename the default files before you upload your files to the server so that you can keep them as a backup.

Now load any page in the ProductCart storefront (e.g. *http://www.yourStore.com/shop/pc/home.asp*, where “http://www.yourStore.com” is your Web site address and “shop ”is the renamed“ productcart folder[1]). Is the page loading correctly? If not, go through the following troubleshooting steps:

1. Are you receiving an ASP error? Make sure the code has been copied and pasted properly. Review the Second and Third steps above for more information. Do not copy and paste the code directly from this document into your HTML template.

[1] If you rename the *productcart* folder, make sure to change the corresponding variable inside the file include/productcartFolder.asp. Download the file from your Web server via FTP, open it in Notepad or your favorite HTML editor, edit it, save it, and re-upload it to your server. Do not remove the *productcart* folder and do not remove or rename the *pc* folder.

2. Are images missing? Make sure that you have altered the links to all image files as mentioned under First Step: create a template.
3. Are links to other files broken? Make sure that you have altered the links to account for the fact that you are two levels down from the root. Also make sure that links to other shopping cart pages point to the same directory. For example, a “Search” link should simply point to “search.asp” since the file is in the same folder (`<a href=search.asp>Search</a>`)

Editing the default header & footer

Creating a professional-looking graphical interface for your store might be time consuming. Some ProductCart users that wish to get their store up & running immediately use an “interim” solution, which consists of using an edited version of the default *header.asp* and *footer.asp* files that ship with ProductCart. You can download them using the following link (the ZIP file also includes the CSS file used to “style” the default store interface):

http://www.earlyimpact.com/productcart/support/ProductCart_v3_header_footer.zip

Follow these simple steps to create a file that you can edit:

- Open *footer.asp* in your favorite HTML editor
- Switch to “Code View” (or alike) and highlight every line of code. You can do so by pressing CTRL+A on your keyboard. Copy the selection (you can do so by pressing CTRL+C on your keyboard).
- Now open *header.asp* and switch to “Code View”. Place the mouse cursor at the very bottom of the file.
- Paste the code from *footer.asp*.
- Save the file with a new name (e.g. header_footer.asp).
- You can now switch to “Design View” and you will see the entire page. Make your edits until you are ready to re-create the *header.asp* and *footer.asp*. Before you re-create the files, make sure that the ASP code mentioned under Second Step: add the ASP code is still there.
- Upload the edited files to the *pc* folder. If you have any problems with the files, make sure to review the troubleshooting steps listed above.

Dynamic Meta Tags

In order to make your Web store more search engine friendly, the code that you have added to *header.asp* will take care of automatically creating the Title, Description, and Keyword Meta Tags for every page in the storefront.

Meta Tags certainly do not carry the kind of importance that they used to have years ago. Today most search engines ignore the Keywords Meta Tag, and - according to most experts – Google's page ranking algorithm ignores both the Keywords and the Description Meta Tags. Yet, Meta Tags still play a role in your overall search engine optimization strategy. For example, experts agree that the Title Meta Tag is still crucial to good page rankings.

Imagine an e-commerce store that has 500 categories and 5,000 products: it would be impossible for the store manager to manually add Meta Tags to all of them one by one. ProductCart will automatically generate them based on product and category information available in the store catalog. Yet, the store manager also has the ability to manually enter them for any product or category.

Let's take a closer look at this feature and see how you can take advantage of it. First, confirm that your *header.asp* file contains the following code towards the top of the page. If not, go back to Second Step: add the ASP code.

```
<!--#include file="include-metatags.asp"-->
<html>
 <head>
  <%if pcv_PageName<>"" then%>
   <title><%=pcv_PageName%></title>
  <%end if%>
  <%GenerateMetaTags()%>
```

These are the lines of code that deal with the dynamic generation of Meta Tags. Highlighted in bold is the name of the file that is "included" in *header.asp* and that takes care of this task for you. The file is called *include-metatags.asp* and is found in the *pc* folder.

Meta Tags are generated as follows:

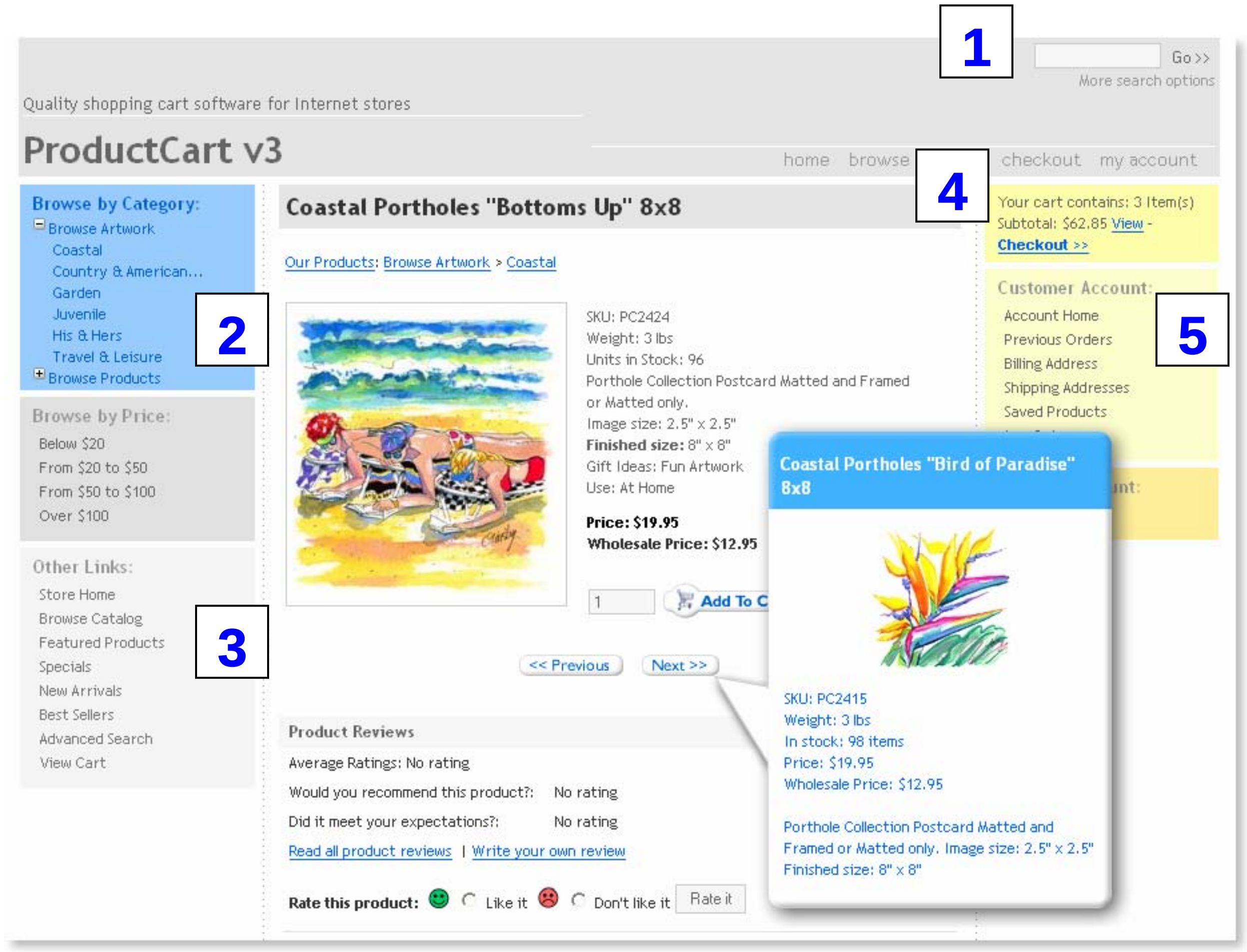
- **Store Manager-Specified Meta Tags**. If Meta Tags have been specified for a product or a category using the ProductCart Control Panel (when adding or modifying a category or a product), then that information is used on the corresponding product or category page. For customers that are upgrading to ProductCart v3: this is a new feature in version 3.
- **Automatically Generated Tags**. If Meta Tags have not been specified for a product or a category, ProductCart automatically generates them for those products or categories, using the category/product name for the Title, and a portion of the category/product description for the Description tag. If you have thousands of products and therefore are unable to manually enter optimized Meta Tags for each of them, this feature provides a sensible solution.
- **Default Meta Tags**. For pages other than product and category pages (e.g. "View shopping cart", "Search", etc.), ProductCart uses the default Meta Tags that are found in the file *pc/include-metatags.asp*. Make sure to edit this file before launching your store to replace the default content with yours. Otherwise, you will see that many of your pages carry the "ProductCart shopping cart software" *Title* tag. If that is the case, it means that the default tags still need to be edited. To edit this file, do the following:
    - Download *pc/include-metatags.asp* to your desktop using your favorite FTP program.
    - Open it in Notepad or an HTML editor, and edit the default tags at the top of the page.
    - Save it and upload it back to your Web server.

Additional Integration: Adding Store Elements to the Interface

To further integrate your Web store's graphical interface with ProductCart, you can add a number of static and dynamic elements to *header.asp* and *footer.asp*. By static, we mean an interface element that doesn't change depending on what the user or the store manager does (e.g. a search box). By dynamic, we refer to an interface element that does change.

Take a look at the picture shown below: this demo store uses the default *header.asp* and *footer.asp* files that come with ProductCart. These files contain a number of these interface elements. Let's look at what they are and how you can use them in your version of header.asp and footer.asp.

1. **Search box**. You can let your customers run a keyword search from anywhere on your site.
2. **Category Navigation**. You can use the Control Panel to dynamically generate a category tree that is then shown in your storefront.
3. **Content Pages and Other Links**. You can automatically show a link to any page generated via the "Manage Content Pages" feature, and use the "Generate Links" feature to create static links to other pages (e.g. New Arrivals, Best Sellers, etc.).
4. **Shopping Cart Total**. You can automatically display the shopping cart total, when something has been added to the shopping cart. The cart summary is hidden if the cart is empty.
5. **Customer Login/Account Links**. You can automatically switch between a login link and an account menu (shown once customers have logged in). You can do the same with Affiliates.



Browse by Category:
Browse Artwork
Coastal
Country & American...
Garden
Juvenile
His & Hers
Travel & Leisure
Browse Products

Browse by Price:
Below $20
From $20 to $50
From $50 to $100
Over $100

Other Links:
Store Home
Browse Catalog
Featured Products
Specials
New Arrivals
Best Sellers
Advanced Search
View Cart

Coastal Portholes "Bottoms Up" 8x8

Our Products: Browse Artwork > Coastal

SKU: PC2424
Weight: 3 lbs
Units in Stock: 96
Porthole Collection Postcard Matted and Framed or Matted only.
Image size: 2.5" x 2.5"
Finished size: 8" x 8"
Gift Ideas: Fun Artwork
Use: At Home

Price: $19.95
Wholesale Price: $12.95

1 Add To C

<< Previous   Next >>

Product Reviews
Average Ratings: No rating
Would you recommend this product?: No rating
Did it meet your expectations?: No rating
Read all product reviews | Write your own review
Rate this product: Like it Don't like it Rate it

Your cart contains: 3 Item(s)
Subtotal: $62.85 View -
Checkout >>

Customer Account:
Account Home
Previous Orders
Billing Address
Shipping Addresses
Saved Products

Coastal Portholes "Bird of Paradise" 8x8
SKU: PC2415
Weight: 3 lbs
In stock: 98 items
Price: $19.95
Wholesale Price: $12.95
Porthole Collection Postcard Matted and Framed or Matted only. Image size: 2.5" x 2.5"
Finished size: 8" x 8"

Let’s take a closer look at each of these 5 interface elements. If you are looking for other ideas, checkout the ProductCart forums at http://www.earlyimpact.com/forum or visit the Developer’s Corner at http://www.earlyimpact.com/productcart/support/developers.asp.

Adding a Search Box

You can easily display a “search box” in your HTML code by including into *header.asp* the file *SmallSearchBox.asp*, which is located in the *pc* folder. To do so, simply add the following line of code to the section of your HTML code where you want the search box to appear (e.g. a table cell):

```
 <!-- #Include file="SmallSearchBox.asp"-->
```

If you need to **edit** the way the search box is presented in your storefront, either open the file *SmallSearchBox.asp* itself in your favorite HTML editor, or edit the main style sheet used by the storefront (*pcStorefront.css*) and edit the classes that pertain to the search box.

If you decide to edit the file itself, you will see that the code reads as follows:

```
<form action="showsearchresults.asp" name="search" method="get" class="pcForms">
 <input type="hidden" name="pageStyle" value="<%=bType%>">
 <input type="hidden" name="resultCnt" value="10">
 <input type="Text" name="keyword" size="14" value="">
 <input type="submit" name="submit" value="Go &gt;&gt;" id="submit">
  <div class="pcSmallText">
   <a href="search.asp">More search options</a>
  </div>
</form>
```

A couple of quick edits can help you change the way the search box behaves:

- Change the *value* of the *pageStyle* hidden field from *<%=bType%>* (this code is retrieving the default page style used in your storefront) to one of the following values to change the way search results are presented:
  - h = search results are shown horizontally
  - l = search results are shown in a list
  - m = search results are shown in a list, with the ability to add multiple products to the cart
  - p = search results are shown vertically
- Change the *value* of the *resultCnt* hidden field from “10” to any other integer to change the number of results shown on the search results page (e.g. “12”).

Adding Category Navigation

The default version of *header.asp* is organized in 3 columns. The left-side column contains three sections of store links. The first section contains links to the categories that were added to the store catalog using the Control Panel.

How can you easily get those categories to appear in your store interface? It doesn’t get any easier than this. You can add category navigation to your store interface by simply adding the following “include” statement.

```
 <!--#include file="inc_catsmenu.asp"-->
```

That’s it. Just one line of code. ProductCart will take care of the rest. Specifically…

- Using the **Settings > Generate Navigation** feature in the Control Panel, you can create the category tree. You can recreate the tree whenever you add, remove or rearrange your categories, to keep it updated. This is a new feature in ProductCart v3. It provides a huge performance boost on stores that contain a large number of categories and products, as the category tree is not built on the fly whenever a customer visits the storefront, but rather built once in the Control Panel, saved to a text file, and then used over and over again in the storefront, without performing any calls to the database.

- By adding the line of code shown above to the section of your HTML document where you want the navigation to appear (e.g. a table cell), you will tell ProductCart to display the category tree there. You can assign a CSS class to the code that is generated by this feature to maintain control on the way the navigation links are displayed.

How it works

ProductCart will generate two static files to store your category and/or product names and locations: one for retail customers, and one for wholesale customers (this is because the Control Panel contains a feature that allows you to hide some categories from retail customers, but show them to wholesale customers: so the navigation might be different for the two customer types).

These files will be used to generate the navigation. This approach (using static files vs. dynamically generating the category tree every time) is used to eliminate redundant queries to the store database to collect the same information over and over (i.e. navigation links remain the same until you change the category tree, and it is a waste of server resources to generate the category tree on the fly every time it is used in the storefront). This also means that the navigation will not change automatically when you add/edit products and/or categories. Remember to rerun this feature whenever you update your product catalog (e.g. you add a new category).

To Create or Update your navigation

Log into the Control Panel, select *Settings > Generate Navigation*, and choose one of the following three options:

1. <u>Include categories and sub-categories, but no products</u>. Customers will be able to expand/collapse the category tree to see the sub-categories. No product links are shown in the category tree.
2. <u>Include only top-level categories and their products</u>. Only the categories listed in the root are shown in the navigation.
3. <u>Include categories, sub-categories, and their products</u>.

If you choose (2) or (3), you will also need to specify the *Maximum number of products per category*. A "More products..." link is automatically added if there more products in the category than the number specified here.

For added flexibility, you can add a CSS class to the navigation code: it is applied to the table cells that are generated when the category tree is built. Simply enter the name of the class in the corresponding input field.

Adding links to Content Pages and Other Key Storefront Pages

The default version of *header.*asp uses a bit of ASP code to loop up “Content Pages” in the store database and display links to them in left-side column. These pages are generated by using the “Manage Content Pages” feature in the Control Panel, which is a basic content management system included in ProductCart. For example, you can use ProductCart to create a “Customer Service” or “Frequently Asked Questions” page right from within the Control Panel.

Let’s take a quick look at the code needed for showing these links.

```
<% ' Show List of Content pages
sdquery="SELECT pcCont_IDPage,pcCont_PageName FROM pcContents where
pcCont_InActive=0;"
set rsSideCatObj=conlayout.execute(sdquery)
do while not rsSideCatObj.eof
%>
<li><a
href="viewcontent.asp?idpage=<%=rsSideCatObj("pcCont_IDPage")%>"><%=rsSideCatObj("pcCo
nt_PageName")%></a></li>
<%
rsSideCatObj.MoveNext
loop
set rsSideCatObj=nothing
%>
```

This code retrieves from the database the list of active Content Pages, and then creates a list of links to them. As mentioned elsewhere in this document, if you want to use this code, we recommend that you copy and paste it directly from *header.asp* using by opening the file in Notepad or your HTML editor. If you copy it from here, the ASP code might not function properly in your file due to extra characters or line breaks that might be created during the "copy & paste" process.

If you are wondering why the code uses the HTML "unordered list" tag (<li>), the reason is that with CSS you can manipulate a list to work great in a navigation area. Here, a cascading style sheet is used to remove the bullets from the HTML list (<li>). The styles that perform this task are included in the CSS file "pcHeaderFooter.css", which is used by the default *header.asp* and *footer.asp.*

Many stores also link to include a links to pages such as Specials, New Arrivals, etc. You can generate the correct URLs to these pages using the **Generate Links** feature in the Control Panel. For example, some of these links are:

```
<a href="home.asp">Featured Products</a>

<a href="showspecials.asp">Specials</a>

<a href="shownewarrivals.asp">New Arrivals</a>

<a href="showbestsellers.asp">Best Sellers</a>

<a href="search.asp">Advanced Search</a>

<a href="viewcart.asp">View Cart</a>
```

Adding a Shopping Cart Summary

Your customers will appreciate having a quick summary of what they have added to their shopping cart shown on every storefront page, so that they know at all times what the total of their order is at that time.

This useful code snippet shows how many items have been added to the cart, the current order total, a link to view the shopping cart details, and one to start the checkout process.

You can add this dynamic interface element to your store interface by including the following line of code (you will find it in the default *footer.asp* file):

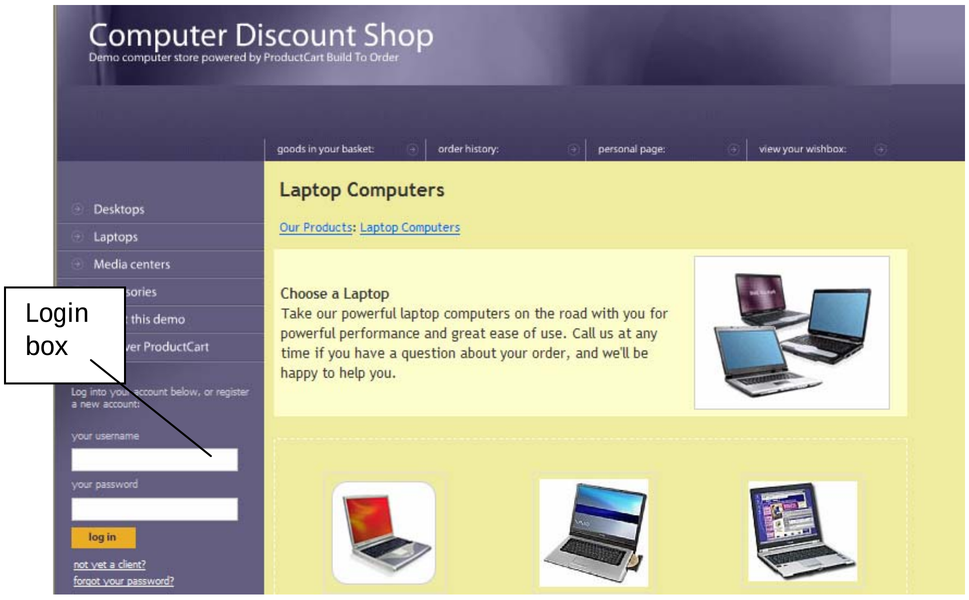
```
<!--#include file="SmallShoppingCart.asp"-->
```

Adding Customer Login/Account Menu

By adding some ASP code to your store interface, you can show customers different information when they are logged into their account vs. when they are not. If they are not logged into their account (e.g. they are not yet a customer), you can show a simple form where they can login and link to register a new account. If they are logged in, you can show a list of links to different areas of their account section.

Let’s take a look at the code found on the default version of *footer.asp*.

```
  <ul>
   <%
   ' If the customer is not logged in
   ' show a link to the registration page.
   if session("idCustomer")="0" or session("idCustomer")="" then
   %>
    <li><a href="custPref.asp">Register/Login</a></li>
   <%
   else
   ' Otherwise show the customer service links
   %>
    <li><a href="custPref.asp">Account Home</a></li>
    <li><a href="CustviewPast.asp">Previous Orders</a></li>
    <li><a href="login.asp?lmode=1">Billing Address</a></li>
    <li><a href="CustSAmanage.asp">Shipping Addresses</a></li>
  <%
  ' If the Wish List feature is active, show a link to it
    if (scWL="-1") or ((scBTO=1) and (iBTOQuote=1)) then
  %>
   <li><a href="Custquotesview.asp">Saved Products</a></li>
  <% end if %>
   <li><a href="CustLO.asp">Log Out</a></li>
<% end if %>
 </ul>
```


Computer Discount Shop
Demo computer store powered by ProductCart Build To Order
goods in your basket:
order history:
personal page:
view your wishbox:
Desktops
Laptops
Media centers
sories
this demo
ver ProductCart
Log into your account below, or register a new account
your username
your password
log in
not yet a client?
forgot your password?
Laptop Computers
Our Products: Laptop Computers
Choose a Laptop
Take our powerful laptop computers on the road with you for powerful performance and great ease of use. Call us at any time if you have a question about your order, and we'll be happy to help you.
Login box

Instead of using a link to the login page, you could also use “**login box**”, as shown in the image above. A login box could be placed on *header.asp, footer.asp,* or even on other Web site pages. We are referring to a small form that contains a user name field, a password field, and a login link or button.

You can wrap a conditional statement around it so that it is only shown when a customer is not logged in, as shown above with the link to *Register/Login*.

A login box in a store running ProductCart v3 should contain the following code:

```
<form name="login" method="post" action="checkout.asp?cmode=1">
   User Name <input type="text" name="LoginEmail">
   Password <input type="password" name="LoginPassword">
   <input name="PassWordExists" type="hidden" value="YES" >
   <a href="JavaScript:document.login.submit();"><img src="images/login.gif"></a>
   <input type="hidden" name="SubmitCO.y" value="1">
</form>
```

If you are upgrading from a previous version of ProductCart, please note the following changes compared to the code that is currently on your page (if any):

- "PassWordExists" is a new field and needs to be passed with a value of "YES"
- The login form should point to "checkout.asp?cmode1" instead of "custva.asp"
- The field names are now "LoginEmail" and "LoginPassword" instead of "email" and "password".
- The "SubmitCO.y" field should remain the same.

As mentioned above, you can use a conditional statement (an "if" statement) to show the login box/links or the account menu links. The conditional statement looks to see if a customer is logged in, using the *idCustomer* session variable.

If a customer is not logged in, a link to register or log in is shown. If a customer is logged in, a few links to frequently used account management pages are shown instead. The link to "Saved Products" is contained in another conditional statement that looks to see whether the Wish List feature is active, or whether the store is using the Build To Order version of ProductCart and customers are allowed to save quotes to their account.

The code uses a list (<li>) to show the links. The list is styled using the pcHeaderFooter.css style sheet.

ProductCart and CSS

Overview

New in ProductCart v3 is the use of cascading style sheets throughout the storefront. Virtually all <FONT> tags where removed and replaced with a number of different classes. Styles were also added to many other elements of the interface.

All styles are stored in a file called **pcStorefront.css**, which is located in the *pc* folder. The file has been heavily commented to help you understand how it affects the look and feel of your store.

All ProductCart users will be able to easily apply basic interface changes to their stores (e.g. setting the font family to "Verdana" or the font color to a certain shade of blue), without needing to be proficient in the use of cascading style sheets. How? All of the interface properties that were previously set through the Control Panel's Display Settings page (e.g. the main font type and color) have been turned into CSS styles and are grouped together at the top of *pcStorefront.css*, as mentioned below.

Web designers and other advanced users will find that this new version of ProductCart gives them a much larger degree of flexibility in altering the look of their stores, thanks to the use of CSS.

First, let's make sure that the cascading style sheet used by ProductCart is correctly loaded in your store.

Loading the style sheets

The first line of code in *header.asp* must always be the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

This line is very important: it defines the type of document that you are asking a browser to render and it also tells the browser how to interpret the cascading style sheets used on the store. Unless this statement is present – and correctly positioned at the very top of your header.asp file – the cascading style sheets used by ProductCart will not function properly.

Next, make sure that pcStorefront.css is loaded in your *header.asp* file by verifying that the following line of code appears before the closing *</head>* tag.

```
<link type="text/css" rel="stylesheet" href="pcStorefront.css" />
```

pcStorefront.css

Let's now take a quick look at the top part of the CSS file to understand how basic interface changes can be made to your storefront.

The styles included under "MAIN Styles" are the ones that used to be controlled by the *Display Settings* area of the ProductCart Control Panel in previous versions of the software. These are the only styles that you will need to edit if you want to change the type of font used in the storefront, its size, its color, etc. Unless you want to, you DO NOT need to change any of the other styles.

```
#pcMain {
	font-family: "Trebuchet MS", Verdana, Arial, sans-serif;
	font-size: 11px;
	color: #333333;
	text-align: left;
	background-color: #FFFFFF;
	width: 100%;
}
```

This first section of the CSS file sets most of the store-wide interface properties that most users will need to edit. Any CSS editor will help you easily change the font family, size, color, etc. It is important that you use a CSS editor rather than trying to edit the file manually as CSS is very "picky" when it comes to interpreting the syntax of the file.

```
#pcMain a:link, a:visited {
	color: #0066FF;
}

#pcMain a:hover {
	color: #0000FF;
	text-decoration: none;
}
```

This other section of the file sets the color for store links. This is also something that was set using the Control Panel in previous versions of ProductCart.

By using CSS, you now have much more flexibility in setting the appearance of links. For example, you could edit the CSS document to specify a different appearance for all different states that a link can be in.

CSS resources

If you are interested in learning more about Cascading Style Sheets, Wikipedia's section on the topic can be a good start. Here are this and other resources:

- http://en.wikipedia.org/wiki/Cascading_Style_Sheets
- http://www.w3schools.com/css/default.asp

Your favorite HTML editor will allow you to properly edit a *pcStorefront.css*. If you don't own an HTML editor such as Dreamweaver® or FrontPage® and are looking for one, here are a couple of inexpensive options:

- http://www.blumentals.net/rapidcss/?refid=g1
- http://www.highdots.com/css-editor/
- http://www.newsgator.com/NGOLProduct.aspx?ProdID=TopStyle