**Online Shopping With Carts**

The most basic function in your Web store is to allow customers to browse the online catalog of products, select items, and place them in their shopping cart. While this seems like a simple process, there's a lot going on in the background that you need to worry about. First, let's look at how to present the store catalog.

**Displaying the Catalog**

So far in our project, you've seen that the navigation section on the main Web page lists the different categories used in the online store. This helps break down the catalog into manageable sections for both you and your customers.

Each category link in the navigation page directs the customer to a Web page that displays the products within that category using the *buyproducts* content page:

echo "<a href=\"index.php?content=buyproducts&cat;=$catid\">$name</a> ($total)<br>\n";

The beauty of using functions is being able to use the same code in multiple places. The *buyproducts.inc.php*file utilizes the same *showproducts()* function you used in the administration back-end application to create a table of the products for the specific category. Let's create the buyproducts.inc.php program:

Create the file *buyproducts.inc.php* in the store folder.

1. Open the file in a text editor, and add the following code:

<?php

$catid = $\_GET['cat'];

$query="SELECT name from categories WHERE catid = $catid";

$result = mysql\_query($query);

$row=mysql\_fetch\_array($result, MYSQL\_ASSOC);

echo "<h2>{$row['name']} - Click on a product to purchase it</h2>\n";

if (!isset($\_GET['page']))

$page = 1;

else

$page = $\_GET['page'];

showproducts($catid, $page, "index.php?content=buyproducts", "index.php?content=updatecart");

?>

1. Save the file and exit the editor.

There's not too much to this code. The link in the navigation section passes the category ID value to the buyproducts.inc.php code. The code takes that value and finds the actual name of the category in the categories table and then displays the category name on the top of the main section. Next, it checks the HTML variable page to see if the showproducts() function passed on the next page value. If not, it assigns the variable to the value of 1 to display the first page.

Finally, it calls the same showproducts() function we used in our administration back-end application to display the products on the main Web page.

A sample catalog Web page

The showproducts() function specifies the two URLs required for the product list. The first URL points to the original Web page for displaying more pages of data. The second URL points to the Web page used when a customer selects an individual product. This code uses the URL:

index.php?content=updatecart

When you click the link for a product, it automatically adds the product ID to the end of this link and sends the customer to the index.php page using the updatecart.inc.php file as the content for the main section.

**Adding Items to the Cart**

The updatecart.inc.php file produces a simple HTML form for the customer to select the quantity of the product to purchase.

The Update Cart Web page

Let's create the code to build this Web page:

1. Create a file called *updatecart.inc.php* in the store folder.
2. Open the file in an editor, and enter the following code:

<?php

$prodid = $\_GET['id'];

echo "<h2>Add item to cart</h2>\n";

$query = "SELECT description, price from products where prodid = $prodid";

$result = mysql\_query($query);

$row=mysql\_fetch\_array($result, MYSQL\_ASSOC);

$description = $row['description'];

$price = $row['price'];

$quantity = 1;

echo "<form action=\"index.php\" method=\"post\">\n";

echo "<table width=\"100%\" cellpadding=\"1\" border=\"1\">\n";

echo "<tr><td>Image</td><td>Product</td><td>Price</td><td>Quantity</td></tr>\n";

echo "<tr><td><img src=\"showimage.php?id=$prodid\" width=\"80\" height=\"60\"></td>\n";

echo "<td>$description</td><td>$price</td>\n";

echo "<td><input type=\"text\" name=\"quantity\" value=\"$quantity\" size=\"3\"</td></tr>\n";

echo "</table>\n";

echo "<input type=\"hidden\" name=\"content\" value=\"addtocart\">\n";

echo "<input type=\"hidden\" name=\"prodid\" value=\"$prodid\">\n";

echo "<input type=\"submit\" value=\"Add to cart\">\n";

echo "</form>\n";

?>

1. Save the file and exit the editor.

The code displays the standard product information (the product image, name, and price), along with a form to enter the quantity to purchase. The form uses a default value of 1 to help make it easier for your shopper to select just a single item.

When your customer clicks the *Add to cart* button, the HTML form passes the product ID and quantity values to the index.php page using the POST method. It assigns the *content* HTML variable the value of *addtocart*.

This causes the index.php file to include the addtocart.inc.php file in the main section. This code attempts to add the product to the current list of products in the shopping cart session array variable.

Now let's build the code for that file:

1. Create a file called *addtocart.inc.php* in the store folder.
2. Open the file in an editor, and enter the following code:

<?php

$prodid = $\_POST['prodid'];

$quantity = $\_POST['quantity'];

$query = "SELECT quantity, description FROM products WHERE prodid = $prodid";

$result = mysql\_query($query);

$row = mysql\_fetch\_array($result);

$stock = $row[0];

$description = $row[1];

if ($quantity > $stock)

{

echo "<h2>Sorry, there are only $stock $description left in stock</h2>\n";

echo "<h2>Please select another quantity</h2>\n";

} else

{

if (isset($\_SESSION['cart'][$prodid]))

{

$\_SESSION['cart'][$prodid] += $quanity;

} else

{

$\_SESSION['cart'][$prodid] = $quanity;

}

echo "<h2>Product added to cart.</h2>\n";

}

echo "<a href=\"index.php\">Continue shopping</a><br>\n";

echo "<a href=\"index.php?content=checkout\">Check out</a>\n";

?>

1. Save the file and exit the editor.

The addtocart code retrieves the product ID and quantity values from the HTML form data, and then it performs an SQL query to check the current quantity value listed for the product in the products table. If your customer attempts to purchase more than the quantity in stock, the code displays an error message and asks the customer to select another quantity.

If there's enough of the product in stock, the addcart code adds the quantity value to a session cookie array variable using the product ID as the variable:

$\_SESSION['cart'][$prodid] += $quantity;

If you're new to programming, this is a shorthand way of writing the statement:

$\_SESSION['cart'[$prodid] = $\_SESSION['cart'][$prodid] + $quantity;

The *+=* symbol adds the specified value to the assignment variable.

This solution also solves the problem for when a customer decides to purchase more of the same item. If the product ID specified in the array variable already exists, it adds the quantity to the existing array variable value.

That should take care of placing new items into the shopping cart. Come on over to the next chapter and see how we can track the items in the cart.