CyberCart Pro Internet Commerce System 3.04

Installation Guide

RT Web Design & Consulting
700 12th Street NW Apt B
Albuquerque, NM 87102
Phone: 505-766-7682
Fax: 505-766-9967
rtweb@cybercart.com
http://www.cybercart.com

TABLE OF CONTENTS

CyberCart Pro Internet Commerce System 3.04
Installation Guide

Introduction

The CyberCart Pro Internet Commerce System is a set of Common Gateway Interface (CGI) scripts and Hyper Text Markup Language (HTML) pages that enables the sale of goods through the World Wide Web. Also referred to as a "shopping cart" script, as the experience is much like going shopping in a grocery store. Customers browse through web pages with items for sale, and when they find something they wish to purchase, they indicate the quantity they would like, and push a button to add the item to their "shopping cart." One of the main functions of the script is to keep track of the items the customers adds to their shopping cart. The script allows the customer to continue to browse through different web pages, adding items until they are ready to "check out." CyberCart will then total up their order, collect billing and shipping information, calculate shipping costs, calculate taxes and provide an invoice-like form for final approval. On approval, both the merchant and the customer are sent (by email) a copy of the order. The order is also stored on the server where the merchant can retrieve the order details through a secure web administration page. An optional module also allows real-time credit card processing through Online Analysis' SocketLink Internet Transaction Gateway.

CyberCart is a form-based system, meaning that the item catalog pages are written using standard HTML forms. On submittal, a CGI script processes the forms. This is in contrast to a cgi-based system, where the actual web pages are also generated from a database. There are benefits and drawbacks to both types of scripts, but the form-based system was chosen for CyberCart to allow maximum flexibility of the web catalog pages. A database module will soon be available that will allow database generation of CyberCart pages.

To maintain a customer's session identification, CyberCart does use a "cookie." A cookie is a small file created by the web browser that is stored on the customer's computer. This means that customers must be using a "cookie capable" browser (such as Netscape 2.0 or Microsoft Internet Explorer 2.0 or greater) and not have cookies disabled. Some people are concerned that cookies can be used nefariously to obtain and store personal or sensitive information. The only thing stored in the cookie issued by CyberCart is a unique session identification number - no personal or user information is recorded.

CyberCart is designed for use by Internet Service Providers or web design consultants with an intermediate to advanced level of understand of both HTML and CGI scripts. CyberCart is written in Perl 5.003, but with the exception of the search function, will run on Perl 4.036. Programming experience is not required, although it never hurts.

This installation guide will give you an overview of how CyberCart works and step-by-step instructions for installing CyberCart. Specifically, the following information will be covered:

- A list and descriptions of the files included.
- Step-by-step installation instructions.
- Merchant configuration parameters.
- CyberCart HTML page construction.

Scripts and Files Included

cybercart.pl
The main work horse of the CyberCart. This script handles most of the major functions of the CyberCart system, except for calculation of shipping cost, which is handled by ccship.pl. The shipping routines were pulled out of the main script to increase the speed and efficiency of cybercart.pl. If the unsecure and secure server have different domains, then a copy of cybercart.pl will need to be installed on both servers - the script will then be able to do a "domain transfer" of the order. More details on this feature will be given in both the **Merchant Configuration Files** and **Setting Up CyberCart** sections.

ccship.pl
This script contains the various shipping subroutines used by CyberCart. This file will be used by cybercart.pl when the shipping charges are being calculated and displayed.

CreditCard.pl
A module from the C-span Perl archive that validates credit card numbers. This script DOES NOT process credit cards - it checks for a valid credit card number using a publicly available algorithm.

onanalysis.pl
Optional module that allows real-time credits card processing. In order to use this, you must be set up to use the SocketLink Internet Transaction Gateway. For more information see http://www.onanalysis.com/. You can also sign up through CyberCart if you are having the installation done by CyberCart.

getorder.pl
Script that allows merchants to view, print, and delete received orders. Also provides some logging and statistical order reporting. This script should be set up on a secure server - the getorder.pl script should NOT be used on an unsecure server. The initial login is done through the getorder.html web page (also included).

summary.pl
Optional script to provide additional *referrer* summary reports. See the **Special Features** section for more information on using the referrer function of CyberCart.

ccdown.pl
An optional script for downloading purchased files. See the **Download File Function** for more information.

demo.pl
This is a demo merchant configuration file. You should make a copy of this script (actually a set of configuration variables described in the **Merchant Configuration Files** section). For each merchant, there will be a separate merchant configuration file - this allows multiple merchants to use the same CyberCart script. The appearance, shipping charges, taxes, etc. can be set differently for each merchant.

addpass.pl

A command line Perl program that allows you to add passwords to the merchant password file (.ccusers). Uses the Unix "crypt" command to encode the passwords. Functions the same as htpasswd, which can be found on most Unix/Linux systems. Provided in case htpasswd is not available.

***.csv**
Theses are various shipping tables used for calculating shipping zones and rates for both UPS and USPS shipping rates. If you use the "actual" shipping rate option, you will need these files. See the **Merchant Configuration Files** section for more information on setting the shipping option.

demo.html
A demo CyberCart HTML page that can be used as a template for other catalog pages. This is only an example layout - since CyberCart is a form-based scripts, the layout of the catalog page is up to the web designer. All that is required is the various form tags detailed in **Creating CyberCart Pages**.

ccdown.html
A HTML page that allows customers to download purchased files. See the **Download File Function** for more information.

Installing CyberCart

Steps for Installation

To install CyberCart IP Pro on an NT host, you will need ftp access with read/write permissions or have administrative permissions. If you are not sure whether or not you have this access, please check with your Internet Service Provider. To install this software, you must have a working knowledge of Perl scripts.

1. Determine the URL for scripts. Where you put the scripts and what the URL for the scripts will vary depending on your IP.

   If you have your own domain name and access to a cgi-bin of your own, the address would be:

   http://www.name.com/cgi-bin/cybercart.pl

   If you are not sure of how to address scripts, check with your IP.

2. Place cybercart.pl into your cgi-bin.

3. Open up cybercart.pl in a text editor (such as Wordpad) and edit the following lines as appropriate. DO NOT open the scripts in a webpage editor such as FrontPage.

   Note: In Perl for Win32, you must use two backslashes '\\' to indicate a directory backslash, as the '\' in Perl is a special character.

   $merchant_data = "C:\\Merchant";
   $merchant_data is where your merchant setup files will be located. For each merchant, there will be a separate setup file. This path should be outside the html document directory (/home/user/public_html) to prevent anyone from snooping in your files.

   $mail_loc = "Blat";
   $mail_loc is the name of the command line mail program on your system. Blat is a good, basic mail program (plus its public domain). Blat can be obtained at:
   http://gepasi.dbs.aber.ac.uk/softw/blat.html

   $mail_server = "mail.name.com";
   Domain of mail server.

   $shipping_dir = $merchant_data . "\\Shipping\\";
   $shipping_dir is where the UPS and USPS data files will be kept.

   $domain = "www.name.com";
   Domain of where the script is.

Note: If you're not sure what your full path is, you can use the UNIX pwd command to find out. That will give you the full pathname of the current directory.

4. Change file permissions. Usually the scripts have to be set for read/write/execute permission for the owner. Use the Windows Explorer to change the permissions or set them using the webserver configuration program.

5. Create a directory called Shipping in your root directory for UPS and USPS rate charts if you will be using actual shipping rats.

6. Create a subdirectory in Shipping called Zones for UPS and USPS zone charts.

7. Place the .csv files in the \Shipping directory.

8. You will also need to obtain the proper zone chart file if using actual shipping rates based on the zip code of the merchant. These can be obtained at: http://www.cybercart.com/Shipping/Zones

   The appropriate one is based on the first three numbers of the merchant's zip code.

9. Now you need to create a merchant file and the merchant directories.

10. In your home directory, create a directory called Demo. Demo is the userid for this example merchant. When you set up other merchants, you assign them a short userid. For the scripts to work correctly, you have to create a directory in your root merchant directory with the same name.

11. Change the permissions of this directory to owner read/write/execute/

12. Create three subdirectories in the Demo directory: Data, Orders, and Logs. If using the Download File Function, create a Files directory as well.

13. Place the Demo.pl merchant setup file in your home directory. **Note: The name of this file will be the same as the merchant userid plus the .pl extension** .

14. Edit the Demo.pl file as instructed by the comments or see Merchant Setup File for more specific information.

15. If the script is on a secure server and you want to enable the credit card checking routine, place the CreditCard.pl file in the home directory where the merchant setup files are kept.

16. Create a directory called Demo in your home html directory.

17. Place the demo.html file in your that Demo directory.

18.Edit the demo.html file by changing the script URL as appropriate. See Creating Pages for more specific information.

19.Change the permissions of the Demo/demo.html file to group/others read/execute permissions.

20.If you are running the script on a secure server, then you will also need to install the getorder.pl script, which allows merchants to retrieve orders through a secure web page. If the script is not running on a secure server, you can still use the getordernt.pl script, but this is not recommended, as data is not encrypted during viewing.

20a.Edit the getorder.pl script:

$home_dir = "c:\\Merchant\\";
Home directory for Merchant files. Note the double '\' on the end.

$script_url = "https://www.name.com/cgi-bin/getorder.pl";
Full URL for the script.

$summary_url = "http://www.name.com/cgi-bin/summary.pl";
Full URL for the summary.pl script.

20b.Place this script in cgi-bin.

20c.Change the permissions to allow the script to run:

20d.Edit the getorder.html page to access the getorder.pl script:

<!-- Edit the URL in this line -->
<form method=post action="https://www.name.com/cgi-bin/getorder.pl">

20e.Place the getorder.html in an html directory.

21.Place the summary.pl script into your cgi-bin directory and change file permissions as for the other scripts.

22.Edit the summary.pl script:

$merchant_dir = "c:\\Merchant\\";
Location of merchant file directory.

$Admin = "userid\@domain.com";
Email address of CyberCart Administrator for error message.

23.Place the ccship.pl file in the directory defined in $merchant_dir.

24. If using Online Analysis real-time credit card processing, place the onanalysis.pl file in the directory defined in $merchant_dir.

25. If using the file downloading feature, place the ccdown.pl in the cgi directory and edit the file. See the **Download File Function** for more information.

Summary of Directories and Files

That's it! To give you an overview, here is what the directory structure should now look like, and where each file should be. This is assuming that your root directory is C :\Merchant, that html files are kept in C:\Inetpub\wwwroot\, and scripts are kept in C:\Inetpub\scripts.

C:\Merchant\Demo.pl
C:\Merchant\CreditCard.pl
C:\Merchant\ccship.pl
C:\Merchant\onanalysis.pl (optional)
C:\Merchant\ccusers.dat
C:\Merchant\Demo\Data\
C:\Merchant\Demo\Orders\
C:\Merchant\Demo\Logs\
C:\Merchant\Shipping\wwwzone.csv
C:\Merchant\Shipping\eeezone.csv
C:\Merchant\Shipping\Zones\004.csv
C:\Inetpub\wwwroot\getorder.html
C:\Inetpub\wwwroot\Demo\demo.html
C:\Inetpub\scripts\cybercart.pl
C:\Inetpub\scripts\getorder.pl
C:\Inetpub\scripts\summary.pl
C:\Inetpub\scripts\ccdown.pl (optional)

Potential Problems

1. Scripts won't run. Make sure you change the permissions. If read/write/execute owner only (700) doesn't work, try adding read/execute group/other permissions.

2. Can't find scripts. Make sure the url for the scripts are correct.

3. Can't open file. Make sure all the required files are in the correct locations.

4. Can't open merchant file. Make sure to include a:

   <input type=hidden name=merchant value=userid>

   tag on all forms.

5. Malformed header errors. You can check the syntax of the perl scripts using the following command:
   perl -c cybercart.pl

   This will compile the script and check for syntax errors without running the scripts.

6. After adding an item, the script doesn't return to the right page. Make sure you have a:

   <input type=hidden name=page_name value="http://www.name.com/page.html">

   tag on the add form. This is the page the script will go to after adding an item.

Addition Assistance

If you continue to have trouble installing the scripts, contact support@cybercart.com for help.

Merchant Configuration Files

Merchant setup files allow each merchant to have a different CyberCart configuration. In addition to the setup file, several directories for storing ordering information must be created. All the merchant configuration variables are in the demo.pl file, with most of them being commented out with a '#'. In Perl, any line beginning with a '#' is viewed as a comment and is not executed. To set a variable, remove the '#' at the beginning of the line. After you set up the merchant file, you need to create the catalog pages.

The demo.pl file is a fully documented example of a merchant configuration file. Make changes to it and save it using the merchant's id followed by .pl as the name. Example, for a merchant with the merchant id of magic, you would save the file as magic.pl and place it in the $merchant_dir directory.

Example Merchant Configuration File

```
##########################################################################
#
# CyberCart Pro Internet Commerce System, Version 3.04
# Copyright 1997, Richard  Torzynski
# 1-25-98
# All rights reserved
# This is NOT a shareware script.
#
# Merchant Setup File for demo merchant on NT
#
##########################################################################
#
# For each merchant to use  CyberCart IP Pro, this merchant file
# defines html page characteristics and also where files are stored
# on the system.
#
# For example, this file is set up for a merchant with the merchant
# ID of demo. This file should be called demo.pl and placed in the
# merchant file directory defined in the main  CyberCart script.
# In this merchant file directory there should be a subdirectory
# called demo with three subdirectories (Data, Orders, and Logs). So here's
# a summary of the directory and file structure:
#
# c:\Merchant
# c:\Merchant\Demo.pl
# c:\Merchant\Demo
# c:\Merchant\Demo\Orders
# c:\Merchant\Demo\Data
# c:\Merchant\Demo\Logs
#
```

```
# These directories should be OUTSIDE of the html root directory so people
# can't browse these files.
#
# CyberCart Script Variables:
#
# Script URL. Set to the full  url of the cybercart script. If you have
# a virtual domain with your own  cgi-bin, it will be:
$script_url="http://www.name.com/cgi-bin/cybercart.pl";
#
# Secure URL. The full URL to the secure server script. Usually this
# means adding an s to http to indicate a secure server is being used.
# $secure_url = "https://www.name.com/cgi-bin/cybercart.pl";
#
# NEW
# Secure Domain.  If the domain name of the secure server differs from the
# unsecure server, then set this variable.  This will enable a domain
# transfer, where the  session_id and order will be 'transferred' to the
# secure server.  This allows you to run the script off of two servers, but
# you must duplicate the directory structure and files onto the secure
# server if its on a separate machine.
#$secure_domain = "www.name2.com";
#
# Toggle for whether script is running on secure server or not.
# 1=secure, 0=not secure
$secure=0;
#
# To supress unsecure ordering feature, set $no_unsecure=1.
# $no_unsecure = 1;
#
# To supress fax/email option, set $no_offline=1.
# $no_offline = 1;
#
# New Mail Error feature.  To help with debugging and possible browser
# related incompatabilities, you can set a toggle to email you when
# someone encounters an error.  This error report will include the date,
# the error message, the users browser,  ip number, the  url, the server,
# and the session_id variables.  This will also be helpful if you need
# to send a message to support@cybercart.com.
$mail_error = 1;

##########################################################################
#
#
# Merchant Data File Variables:
#
```

```
# Path of directory of merchant data directory
$home_dir="C:\\merchant\\Demo";
#
# Order Directory - temp order files stored here.
$order_dir= $home_dir . "\\Orders";
#
# Invoice Directory - Final invoices stored here.  getorder.pl script is
# used to view these files using  netscape or MSE.
$invoice_dir= $home_dir . "\\Data";
#

# Log files are kept here.
$log_dir = $home_dir . "\\Logs";

# Status File - stores order status information.
$status_file= $home_dir . "\\status.dat";

#
# Client list file.
$clientlist = $home_dir . "\\referrer.dat";
$sendclient = 1;
#
# Don't change this.
$price_file = "none";
#
#
##########################################################################
#
# Merchant HTML File Variables:
#
# URL of directory where merchant HTML files are. This can be on another
# system or provider. End this with a slash.
$home_url="http://www.name.com/Demo/";
#
# URL of merchant home page.
$home_page="http://www.name.com/Demo/demo.html";
#
# Full Path name of merchant html files (if stored on this system). Used
# by search engine. If files are stored on another server, you need to
# install the cybersearch.pl script on that server to use the search engine.
$home_html = "c:\\inetpub\\wwwroot";
#
#
##########################################################################
#
```

```
# Company Information
#
# Company email address. This is where orders are   emailed to.  Make sure you
# escape the '@' by putting a '\' in front of it.
$recipient="userid\@name.com";
#
# Company Name
$company="Company Name";
#
# Company Phone Number
$phone="1-999-999-9999";
#
# Company Fax Number
$fax = "1-999-999-9999";
#
# Address
$address="123 Main Street, City, State, Zip";

# Additional header info printed underneath the company info
# $additional_header = " ";

#
# Set to "none" if don't use  creditcards
# $creditcards = "none";

# Payment types accepted (list  creditcards, checks,  moneyorders, etc.).
@credit=("Check","Visa","Mastercard");
#
# Checks. Who checks should be made out to .
$checks="Company";
#
# State taxes.  Us the  two letter  appreviation for states.
%statetaxes=("NM",".0525");
#
# Tax option.  In some states (such as California), shipping and
# handling is taxable.  Set this variable to 1 to add shipping and
# handling to taxable total.
# $tax_option = 1;
#
# State textbox. Set this to 1 to have a textbox instead of drop down
# menu of states or 2 to  supress state input field for countries
# other than the United States.  If you  supress, remember to remove
# state form @required array down below.
# $state_textbox = 2;
#
```

```
#########################################################################
#
#
# Script Pages  Appearence Variables. Sets the properties of script
# generated forms.
#
# title pict that appears at the top.
# $titlepict="http://www.name.com/logo.gif";
#
# Button images.  If you wish to use images instead of standard
# form buttons, set each of these to the URL of the image.
#
# Directory of where the button images are located
# $button_dir = "http://www.name.com/Demo/graphics/";
#
# Button images
# $button_search = $button_dir . "buttonsearch.gif";
# $button_order= $button_dir . "buttonorder.gif";
# $button_check= $button_dir . "buttoncheck.gif";
# $button_clear= $button_dir . "buttonclear.gif";
# $button_shop= $button_dir . "buttonshop.gif";
# $button_continue= $button_dir . "buttoncontinue.gif";
# $button_update= $button_dir . "buttonupdate.gif";
# $button_secure= $button_dir . "buttonsecure.gif";
# $button_online= $button_dir . "buttononline.gif";
# $button_offline= $button_dir . "buttonoffline.gif";
#
# Body background (specify full  url of background image)
$bodyback = "";
#
# Body background color.
$bodycolor = "FFFFFF";
#
# Body text and link colors.
$bodytext = "";
$bodylink = "";
$bodyvlink = "";
$bodyalink = "";
#
# Define table background tables
$Table_width = "580";
$Table_Header_Color = "C9CFF8";
$Table_Body_Color = "FFFFE1";
$Table_Entry_Color = "C8FFC8";
#
```

```
# Item properties used by merchant (Size, Color,  Style). Can have up to
# three.
@properties=("Color");
#
# Pagelinks that appear at the bottom of each script generated page. The location
# specified in the first of the pair, is the file location relative to the
# $home_url specified directory.
%pagelinks=("demo.html","Demo Book Page");
#
# List of required information in order form.
# Possible choices:  Name,Street,City,State,Zip,Country,Phone,Email .
@required = ("Name","Street","City","State","Zip","Country","Email");
#
# Show order toggle.  Set this variable to 1 to show contents of cart after
# customer adds item to cart.
# $showorder = 1;
#
# Redirect after adding.  Set this variable to the number of seconds to
# wait before returning customer to last page after adding item to cart.
# Comment this variable out if you don't want them redirected back automatically.
# $redirect=3;
#
# Set $print_code = 1 to print the order number of the items on the forms
# $print_code = 1;
#
# Message to put at the end of the customers receipt
$mail_closing = "\n
Thank you for ordering from $company! If your order as listed is
incorrect, please let us know immediately!
Email: $recipient
Phone: $phone";
#
# Order Note.  General message printed out in the order form.  Frequently used to
# indicate that shipping will be calculated later for international orders.
# $order_note = "Order Note";
#
#########################################################################
#
#
# Shipping variables.
#
# Method of calculating shipping [ flat,percent,table,weight,items,actual,none ]
# Select one of these methods, uncomment the lines and set variables
# needed for that method.
#
```

```
# Base country for shipping charges.
$shipping_base = "US";
# ================================================================
# Shipping Method 1
# For flat, set the $shipping_modifier to the flat shipping charge.
$shipping_cost = "flat";
$shipping_modifier = 4.5;
$world_shipping_cost = "flat";
$world_shipping_modifier = "10";
#
# ================================================================
# Shipping Method 2
# For percentage of the total cost.
# $shipping_cost = "percent";
# $shipping_modifier = .10;
# $world_shipping_cost = "percent";
# $world_shipping_modifier = .10;
#
# ================================================================
# Shipping Method 3
# For a table based on the total cost.
# Associative array containing ranges, and the cost of shipping for each of
# the ranges.
# $shipping_cost = "table";
# %ship_table=("1-20","3.00","20-50","4.50","60-100","6.00","100-300","10.00","300-
1000","15.00");
# $show_table = qq[
#  1 -  19.99     \$ 3.00
# 20 -  49.99    \$ 4.50
# 50 -  99.99    \$ 6.00
#100 - 299.99  \$10.00
#300+             \$15.00
#];


# $world_shipping_cost = "table";
# %world_ship_table =("1-20","13.00","20-50","14.50","60-100","16.00","100-
300","20.00","300-1000","25.00");
#$world_show_table = qq[
#  1 -  19.99    \$13.00
# 20 -  49.99    \$14.50
# 50 -  99.99    \$16.00
#100 - 299.99    \$20.00
#300+           \$25.00
#];
```

```
#
#
# ================================================================
# Shipping Method 4
# For a table based on the total weight. If you use this method, each item
# must contain a hidden form variable for weight,
# <input type=hidden name=id_weight value=itemweight>
# $shipping_cost = "weight";
# %ship_table=("1-2","2.00","2-4","4.00",4-10","6.00","10-20","8.00");
# $world_shipping_cost = "weight";
# %world_ship_table=("1-2","2.00","2-4","4.00",4-10","6.00","10-20","8.00");
# Measurement units weight is given in if using weight to calculate shipping.
# $weight="lbs";
#
# ================================================================
# Shipping Method 5
# For a table based on the number of items ordered.
# $shipping_cost = "items";
# %ship_table=("1-2","3.00","3-4","4.00","5-6","6.00","7-10","10.00");
# $world_shipping_cost = "items";
# %world_ship_table=("1-2","2.00","2-4","4.00",4-10","6.00","10-20","8.00");
#
# ================================================================
# Shipping Method 6
# To use UPS and USPS actual shipping rates from within the United States.
# $shipping_cost = "actual";
# $world_shipping_cost = "actual";
# Set the weight units - this must be set for weight to show up on the order forms
# $weight = "lbs";
#
# If using actual, uncomment and set the variables = 1 for the services
# used by merchant
# $upsground=1; # UPS Commercial Ground
# $ups2day=1; # UPS 2 Day Air
# $ups2dayam=1; # UPS 2 Day Air AM
# $ups1day=1; # UPS Next Day Air
# $ups1daysav=1; # UPS Next Day Air Saver
# $ups3daysel=1; # UPS 3 day Select
# $usps_express=1; # USPS Express Mail
# $usps_priority=1; # USPS Priority Mail
# $usps_mparcel=1; # USPS  Machinable Parcel
# $usps_nmparcel=1; # USPS Non- Machinable Parcel
#
# Merchant zip code for determining domestic shipping rates
# $merchantzip = "87106";
```

```
#
# Merchant State for determining world UPS shipping rates
# $merchantstate = "NM";
#
# Path of appropriate shipping zone chart.  The zone chart is based
# on the first three digits of the *merchants* zip code that the order
# is being shipped from.
# $zone_chart = "c:\\merchant\\Shipping\\Zones\\870.csv";
#
# ================================================================
# Shipping Method 7
# Special per item shipping cost.  If the shipping cost varies by
# item, you can set the shipping charge individually.  On the html
# pages, you must include an additional hidden form variable
# called item_ship.  See the pages.html for more instructions.
#
# $shipping_cost = "per_item";
#
# ================================================================
# Shipping Method 8
# Custom shipping table for international orders.  Special tables
# must be created to use this method - see specialship.html for more
# information.
#
# Tables can be constructed to determine rate by either weight or
# number of items.
# Special shipping by weight
# $shipping_cost = "special_weight";
# $shipping_cost = "special_items";
# $world_shipping_cost = " special_weight";
# $world_shipping_cost = " special_items";
#
# Path name of the special zone chart.
# $special_zone = "c:\\merchant\\Shipping\\State_Zone.txt";
# $world_special_zone = "c:\\merchant\\Shipping\\Country_Zone.txt";
#
# Path name of the special rate chart.
# $special_rate = "c:\\merchant\\Shipping\\State_Rate.txt";
# $world_special_rate = "c:\\merchant\\Shipping\\Country_Rate.txt";
#
# Name for the shipping method to be displayed on  orderform
# $special_name = "Demo Shipping Charges";
#
# Max item or weight.  The upper limit of the custom charts.
# $max_item_rate = 23;
```

```
#
# Item or weight increment above the $ max_item_rate.
# $item_inc = 5;
#
# Rate per item or unit weight above the $ max_item_rate.
# @add_item = (0,0,3.5,5.5,7.5,9.0);
#
# ================================================================
# Shipping Method 9 - NO SHIPPING COST
# No shipping option. If the price includes shipping, set shipping to none.
# $shipping_cost = "none";
#
# ================================================================
#
# Set this to 1 for no International Orders
# $no_world_shipping = 1;
#
# Set to the minimum shipping cost.  Shipping cost will be set to this minimum
# if calculated shipping is less.
# $min_shipping = 2.00;
#
# Set the maximum shipping cost.
# $max_shipping = 12.00;
#
#####################################################################
#
# Handling costs.
#
# If there is additional cost for handling, indicate one of
# the following options:
# For handling charge for whole order:
# $handling_order = 2;
#
# For handling charge for EACH ITEM:
# $handling_item = 1;
#
# Per item handling.  A special form variable must be included for each item.
# $handling_special = 1;
#
#####################################################################
#
# Discounts Variables.
#
# To apply a discount for all items, or a discount if you order a
# certain amount, set this variable to one of the following:
```

```
# [fixed_table,percent_table, percent, none].
#
$discount = "none";
#
# Fixed_table discount. Set a range of total amount ordered, and the amount of
# the discount.
# $discount = "fixed_table";
# %discount_table = ("0-20",0,"20-50","5.00","50-100","10.00");
#
# Percent_table discount. Set a range of total amount ordered, and the percentage
# discount:
# $discount = "percent_table";
# %discount_table = ("0-10",0,"10-20",".05","20-100",".10","100-500",".15");
#
# Percent. Straight percent of total.
# $discount = "percent";
# $discount_percent = ".10";
#
#
#########################################################################
# Special Options:
#
# Required fields. List of required fields. Options are:
# Name, Street, City, State, Zip, Country, Phone, Email
@required = ("Name","Street","City","State","Zip","Country","Email");
#
# Special Selections. These are options that are added onto the
# billing form. If there are cost associated with them, they will be
# listed in a separate entry on the invoice form.
# The format is:
# $special_selection{'Name'} = "form element | text | additional cost";
# The additional cost will be added to the total amount and shown in
# form as its own entry. The name of the form variable must be the
# same as the key in the $ special_selections. For example, to add cod:
#
# $special_selections{'COD'} = "<input type=checkbox name=COD>|Send by COD|4.95";
#
# Additional Fields.
# Use to include additional fields in the form. Note that the name
# of the hash has to be the same as the "name" of the form element.
# Separate the title and form tag with the '|' character.
# $additional_fields{'Work Number'} = "Work Number|<input name=\"Work Number\" size=20
maxsize=20>";
#
#
```

```
# Minimum total order amount.  Set this to the minimum order amount.  Customer will
# get a warning if their order is less than this minimum amount.
# $min_total = 10;

# Set $no_gift_message to "1" to suppress print of gift  textarea.
# $no_gift_message = 1;

# Set $no_billing_address to "1" to suppress printing of separate billing address option
# $no_billing_address = 1;

# Supress the decimal point in the currency displayed
# $no_decimal = 1;

#########################################################################
# Onanalysis for real-time credit card processing
# See th instruction manual and Online Analysis' Web Site for
# complete instructions on using Online Analysis Gateway.
#
# Enables
# $onanalysis = 1;
#
# Vendor ID assigned by Online Analysis
# $vendor_id = "Demo";
#
# Vendor Password assigned by Online Analysis
# $oa_password = "analyze";
#
# $transaction_type = "C6";
#
# Transaction log location
# $oa_log_file = $home_dir . "\\Logs\\oalog.dat";
#
# onanalysis script references which you get from  onanalysis
# $oa_ref = "/socketlink/demotest.cgi";
#
#################################################################
#
# Download enable
# Each file must be kept in the Files subdirectory of the
# merchant's root directory.  Each item must also have a
# code_dlfile hidden form variable.
#
# There are three methods of download:
# $dl_type = 1 : A list of files will be shown immediately
# after customer completes the order.  Least secure.
```

```
# $dl_type = 2 : An email message is sent to the customer
# with the download authorization code.
# $dl_type = 3 : An email message is sent to the merchant
# with the download authorization code which can then be
# forwarded to the customer after payment processing.
# $dl_type = 2;
#
# URL for the CyberCart Download script
$dl_url = "http://www.name.com/cgi-bin/ccdown.pl";
#
# URL for the Web Page interface where customers can
# enter their download authorization code.
$dl_page = "http://www.name.com/ccdown.html";
#
# Location of the files to be downloaded.  This should be a file
# outside the root directory.
$dl_files = $home_dir . "/Files/";
#
# The merchant_id
$merchant_name = "Demo";
#
##########################################################################
#
# Copyright 1997, Richard  Torzynski
#
##########################################################################

return 1;
```

Merchant Directories

You also need to create several directories in your home directory for each merchant. The getorder.pl script will now do this automatically for you in most cases. Temporary and order files are kept here. For each merchant you need a directory called whatever the merchant id is. For example, if the merchant id is Demo, then there needs to be a Demo directory in your home directory. There should also be a merchant setup file called Demo.pl. Underneath the demo directory, there should be 3 subdirectories: Data, Orders and Logs. So the structure will look like this assuming your home directory is C:\Merchant\ and you web pages are at C:\Inetpub\wwwroot\

C:\Merchant\Demo.pl
C:\Merchant\CreditCard.pl
C:\Merchant\template.pl
C:\Merchant\ccusers.dat
C:\Merchant\Demo\Data\
C:\Merchant\Demo\Orders\
C:\Merchant\Demo\Logs\
C:\Merchant\Shipping\wwwzone.csv
C:\Merchant\Shipping\eeezone.csv
C:\Merchant\Shipping\Zones\004.csv
C:\Inetpub\wwwroot\getorder.html
C:\Inetpub\wwwroot\Demo\demo.html
C:\Inetpub\scripts\cybercart.pl
C:\Inetpub\scripts\getorder.pl
C:\Inetpub\scripts\summary.pl

Adding Merchant to .ccusers Password File

You can use the perl script called addpass.pl, which is available at http://www.cybercart.com/Shipping/addpass.pl. Put this file in the merchant data directory and run the program from the DOS prompt. To create or add a password using addpass.pl:

C:\merchant\>perl addpass.pl merchantID .ccuser

Creating CyberCart Pages

To create item pages for the shopping cart script, you use hidden form tags to enable the different script functions. You can format the pages anyway you like as long as you include the necessary hidden tags.

Add Item Tags

To add an item to customers shopping cart script, you need to include the following elements:

Form Tag

At the top of the page, you will need a <form> tag to send the page for processing. You do not have to use the <form> tag around each item; you only need to enclose all the item information in the one form tag. This also allows several items to be processed at once.

The method is POST and the action is the URL of the script:

<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">

Merchant userid

To know which merchant setup file to use, the Multi-Merchant version script uses a hidden variable to pass the name of the merchant.

NOTE:This form variable is not required on the Single Merchant or Demo Version of Cybercart.

<input type=hidden name=merchant value=userid>

So for the Demo pages, this is:

<input type=hidden name=merchant value=Demo>

Action

This hidden variable determines the action the script will take. For adding items, this is add:

<input type=hidden name=action value=add>

You can also have the script go directly to the checkout process by setting the action variable to addplus:

<input type=hidden name=action value=addplus>

Page_name

This hidden variable determines where the script goes after adding items. Either use a full URL, or a relative path name. If the script is running through a cgi wrapper, such as cgiwrap, then you might have to use the full URL here.

<input type=hidden name=page_name value="demo.html">

Item Variables

Each item will have a number of hidden form variables. Some are required, and some are optional. Here is a description of the different possible hidden variables and when you need to use them.

Name and anchor variables

Name required and anchor optional.
****
<input type=hidden name=id_name value="Item Name">

The id_name should be unique. I usually use a two-letter code followed by a three-digit number and group the items into categories. For example, for a juggling club, it might be:

****
<input type=hidden name=jg002_name value="Dube Airflite Club">

Quantity variable

<input type=text name=id_quant value=0>

The id has to be the same as in the above name variable. This produces a text box where the customer can indicate item quantity. Example:

<input type=text name=jg002_quant value=1>

You can also use a checkbox for quantity:

<input type=checkbox name=jg002_quant value=1>

Or a drop down box:

<select name=jg002_quant>
<option>0
<option>1
<option>2
<option>3
</select>

Minimum and Maximum Quantity variables

<input type=hidden name=id_min value=minimum>
<input type=hidden name=id_max value=maximum>

If included, and quantity selected is less than the minimum, then the quantity is set to the minimum automatically. Likewise, if quantity selected is greater than the maximum, then quantity is set to the maximum automatically. The customer could still get around this by adding the same item twice to go over the maximum. You can also just use one of these for an item - both are not required. Example:

<input type=hidden name=jg002_min value=3>
<input type=hidden name=jg002_max value=12>

Price variable

<input type=hidden name=id_price value=itemprice>

Example for the juggling clubs:

<input type=hidden name=jg002a_price value=13>

The script is also setup to accommodate an item that has slightly different properties, but at different prices. For example, if you have a book that comes in hardcover and softcover, with different prices, you can use a drop down selection box for this, if you format it right.

<input type=select name=id_price>
<option>type1-$price1
<option>type2-$price2
</select>

Where type1 would be the additional item distinction, and $price1 is the price associated with that distinction. In the check out form, the type1 will be added to the name of the item. Note, this is not

the same as using item properties. So, for example, to do a book with hardcover and softcover options:

<input type=select name=book1_price>
<option>Soft Cover-$5.95
<option>Hard Cover-$21.95
</select>

Note:For this to work, you must have the -$ separating the distinction and price. The script keys on these two characters. The price of an item can also be modified by a property as described below.

Quantity Pricing

You can also have different prices for an item based on quantity. To enable this, format the id_price by specifying the range of quantity separated by 'to' and a '!' separating the price for that range, and then a '|' separating these different quantity price ranges. Sounds more complicated then it is, so here is an example:

<input type=hidden name=id_price value="1to10!9.95|11to20!9.50|21to100!9.00">

Note: If the customer specify a quantity that is higher than the last range, then the price is set to that last range.

Non-Taxable Items

To mark an item as non-taxable, add an -nt after the price. To calculate the order total, the script will keep separate totals for taxable and non-taxable items. Tax is applied only to the taxable item total.

<input type=hidden name=id_price value=price-nt>

For a book that is non-taxable:

<input type=hidden name=bk001_price value=9.95-nt>

Properties variables

If items have properties such as color or size that you specified in the @properties array in the merchant setup file, then you need to include a way to indicate property choices. You can use SELECT, TEXT or RADIO form elements for this. Note: The script does not support SELECT MULTIPLE form elements at this time.

Colors <select name=jg002_prop1>
<option>Blue
<option>Green
<option>Pink
<option>Purple

<option>White
<option>Yellow
</select>
Note the "1" at the end of "prop1". This indicates a value for the first property in the @properties array. If you defined @properties=("Color","Size"), then "prop1" would refer to color and "prop2" would refer to the size.

Price Adjustment

The base price of an item (as set by id_price) can be adjusted through the properties. You can add to item's base price by adding a "+" (or "-") and the amount to add or subtract in the value of the form variable. For example, another way to adjust the price of a book would be to use a Type (hardback or softcover) property, with the softcover as the baseprice:

Type <select name=book1_prop2>
<option>Softback
<option value=Hardback+16>Hardback
</select>

The full versions can handle up to three properties, while the demo version allows one property variable.

Weight variable

(optional, but required if you use weight in calculating shipping).

<input type=hidden name=id_weight value=weight>

Again, you only need to include this variable if you are determining shipping and handling cost by total weight. The weight should be in the unit you defined in $weight (lbs, kgs, etc.). Example:

<input type=hidden name=jg002_weight value=.5>

Item Shipping variable

(optional, but required if you set the $shipping_cost variable to "per_item").

<input type=hidden name=id_itemship value=amount>

Example, for a book with shipping cost of $2.00:

<input type=hidden name=book1_itemship value=2>

Item Handling variable

(optional, and can be used if the variable $handling_special is set to 1).

<input type=hidden name=id_handling value=amount>

Example, for a book with an addition $1.00 handling charge per book:

<input type=hidden name=book1_handling value=1>

Download File variable

If the item is a file for download, then you need to in clude this variable specifying the filename. This file must also be placed in the Files subdirectory. Also, make sure the mime-type is correctly specified in the summary.pl file.

<input type=hidden name=id_dlfile value=filename.ext>

Example, for a PDF juggling instruction book:

<input type=hidden name=book2_dlfile value=juggling.pdf>

That's all the hidden variables that need to be defined for the items.

Submit Button

At the bottom you need to include a submit button to add the items. You can also add multiple submit buttons or use an image instead of a form submit looking button:

<input type=submit value=button_name>
</form>

Example:

<input type=submit value="Add Items"> </form>

To use an image as a button:

<input type="image" align=right src="picture.gif" name="action" value="add" border=0>

Other Script Functions

You also need a set of buttons for the script to allow other functions. These can be included on each catalog page (recommended) or can be in a frame somewhere. I usually put them in a table to make their appearance neater. These buttons allow five actions to be taken:

Search feature

This feature is disabled in the demo version. As mentioned above, you will need to have a name anc hor tag at the top of each item for the search engine to be most effective.

```
<form method=post action=script_url>
<input type=hidden name=merchant value=userid>
<input type=text name="term" size=15 maxlength=40>
<input type=hidden name=action value=search>
<input type=submit value="Search">
</form>
```

Show current order

Will show the current order if pressed.

```
<form method=post action=script_url>
<input type=hidden name=merchant value=userid>
<input type=hidden name=action value=order>
<input type=submit value="Show Order">
</form>
```

Check out

When the customer is done shopping and wants to submit the order.

```
<form method=post action=script_url>
<input type=hidden name=merchant value=userid>
<input type=hidden name=action value=place>
<input type=submit value="Check Out">
</form>
```

Clear order

This will clear the customer’s order by deleting the temp file.

```
<form method=post action=script_url>
<input type=hidden name=merchant value=userid>
<input type=hidden name=action value=clear>
<input type=submit value="Clear Order">
</form>
```

Customer Order Status

CyberCart allows customers to check on the status of their orders, and for merchants to change the status displayed to customers. The merchant can put whatever message they like and change it through the Administrative Script (called getorder.pl).

This function can be enabled either through the addition of a new form, or in conjunction with a little JavaScript to make entering the info a little more sophisticated. Below are instructions for this feature using the 2 different options.

Form Only

You need to include 2 textboxes for input. The customer must enter the Name from their order and the order number, which is included in the email receipt they receive. This form could be at the top of each page as one of the function buttons, or you make a special page by itself.

Here is the html code for the form tags:

```
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">
<input type=hidden name="merchant"  value="Demo">
<input type=hidden name="action" value="checkstatus">
Customer Name: <input type=text size=10 name=customer>
Order Number:<input type=text size=10 name=ordernum>
<input type=submit value="Order Status">
</form>
```

JavaScript Enabled Form

Using the following JavaScript code, you can have input boxes pop up to prompt the customer for the information. To enable this, place the JavaScript code in between the header tags, <head> </head>

```
<script language="JavaScript">

function GetStatus (formObj) {
   var promptStr1 = "Name of person who placed order?"
   ans1 = prompt(promptStr1,"");
   var promptStr = "Order Number?"
   ans = prompt(promptStr,"");
   if (ans == null || ans1 == null) return false;
   else {
     formObj.ordernum.value = ans;
     formObj.customer.value = ans1;
     return true;
```

```
   }
}

</script>
```

You also need to place the following form whereever you want the status button to be:

```
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl"
onSubmit="return GetStatus(this)">
<input type=hidden name="merchant"  value="Demo">
<input type=hidden name="action" value="checkstatus">
<input type=hidden name=customer value="">
<input type=hidden name=ordernum value="">
<input type=submit value="Order Status">
</form>
</pre>
```

Changing the Order Status

The merchant can change the status of an order through the CyberCart Administrative script (getorder.pl) by clicking on the "Status" button for the order.

Example Function buttons

Here's an example of the four buttons put into a table that you can cut and paste:

```
<table>
<tr valign=top align=center><td>
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">
<input type=hidden name=merchant value=userid>
Term:<input type=text name="term" size=15 maxlength=40>
<input type=hidden name="action" value="search">
<input type=submit value="Search">
</form>
</td>
<td>
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">
<input type=hidden name=merchant value=userid>
<input type=hidden name="action" value="order">
<input type=submit value="Show Order">
</form>
</td>
<td>
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">
<input type=hidden name=merchant value=userid>
<input type=hidden name="action" value="place">
<input type=submit value="Check Out">
</form>
</td>
<td>
<form method=post action="http://www.name.com/cgi-bin/cybercart.pl">
<input type=hidden name=merchant value=userid>
<input type=hidden name="action" value="clear">
<input type=submit value="Clear Order">
</form>
</td></tr>
</table>
```

Referrer Function

CyberCart has a built in referrer function that allows merchant to track where customers link to their site from and allow the referrer to see a summary of orders resulting from their link. This is for sites that wish to compensate other websites for adding links from their site to the merchant site. This function can also be set to send a copy of the order to the referrer as well.

How It Works

The referrer puts a link to the "enter" subroutine of CyberCart. When someone clicks on the link, a cookie is set identifying the referrer, and then is redirected to the starting page of the merchant. When the customer orders something, CyberCart logs the referrer information, if present. The CyberCart Administration script allows the merchant to view a summary of orders, by referrer. The referrer can also connect to a "summary" script that allows them to see a log of orders that resulted from their link (only the order number, date, and net and gross totals are shown to the referrer).

How to Set it Up

1. The referrer must sign up with the merchant and be added to the referrer.dat file. This is a colon delimited text file containing the referrer's id, email address and password (which is optional). This information can be added either manually, or through the Administration script. Here is an example entry:

   rtweb:rtweb@lobo.net:1234

   If a password is set in this file, then when the referrer connects to the summary, they must include the password in the URL.

2. To use the client function, you have the referrer set a link on their site to the CyberCart script. The format is as follows:

   To Store

   Where merchant is the merchant id of the shopping cart pages, referrer is the name of the referrer. How you have the referrer sign-up is up to you.

   This will send the customer first to the shopping cart script, where they will be assigned a cookie for their session id, and also a cookie for the client information. When they check out, this information will be reported in the final order invoice that the merchant receives.

3. You can also have a short email message containing the order number, date, net amount, and gross amount sent to the referrer as a courtesy. In the merchant configuration file, set the $ sendclient = 1 to enable this.

   $sendclient = 1;

4. Place the summary.pl script in the cgi directory.

5. Modify the summary.pl file.

```
# summary.pl setup variables:
#
# Location of merchant file directories
$merchant_dir = "c:\\Merchant";

# Email address of CyberCart Administrator
$Admin = "userid\@name.com";

# URL of this script
$script_url = "http://www.name.com/cgi-bin/summary.pl";

# Location of the sendmail program
$mail_loc = "c:\\Blat\blat.exe";
```

7. Change the permission of the summary.pl file as appropriate.

Summary Information

The merchant can view a log of orders and whom they were referred through in the CyberCart Administration script (getorder.pl). The referrer can also view a summary of their orders using the summary.pl script. To view a summary of their orders, they link to the site with the following URL:

http://www.name.com/cgi-bin/summary.pl/merchant :referrer:password

where merchant is the merchant's id and referrer is the value they used in the link on their pages as discussed above, and the password for the referrer (optional). This will show the date, order number, amount of the purchases, and a grand total.

Special Shipping Option

To provide more flexibility for calculating shipping cost, CyberCart can now use custom shipping tables similar to those used by UPS and the United States Postal Service. To You can use custom tables for both the base country (defined by the $shipping_base variable with the default being US) and/or other countries.

The special shipping tables can also be by weight or items ordered. If by weight, then the rate will be determined by total weight of all the items (and each item must have a hidden variable called itemcode_weight set), or if by item, then the rate will be determined by the total number of items ordered.

To use the special shipping tables you will also create two files. One, a zone chart that defines what zone each country (or state) is in. Two, a rate chart, that defines the shipping rate for each of the zones listed in the zone chart.

Here are the steps to using special shipping tables:

1. If setting up a special shipping table for the base country, set the $shipping_cost variable to either "special_weight" or "special_items". For other countries, set the $world_shipping_cost to either "special_weight" or "special_items".

2. Define the path name of the zone chart:
   $special_zone = "C:\\Merchant\\Demo\\Cntry_Zone.txt";
   and/or
   $world_special_zone = "C:\\Merchant\\Demo\\Cntry2_Zone.txt";
   Name these files as appropriate.

3. Define the path name of the rate chart:
   $special_rate = "C:\\Merchant\\Demo\\Cntry_Rate.txt";
   and/or
   $world_special_rate = "C:\\Merchant\\Demo\\Cntry2_Rate.txt";
   Name these files as appropriate.

4. Set the name for the special shipping method:
   $special_name = "Demo Shipping Charges";
   This will show up on the script generated form pages.

5. Set the max item or weight. Above this number of items or weight, then the shipping cost can be set incrementally with the next 2 variables.
   $max_item_rate = 23;

6. The item or weight increment when the max_item_rate is exceeded. So, for example, if after the max of 23, an additional cost for each 5 above this could be set.
   $item_inc = 5;

7. The last variable to set is for the rate, and is defined in an array. So if you have 5 zones defined in your zone chart, then you would have an array of 6 (5+1) rate increments set:
   @add_item = (0,2,3.5,5.5,7.5,9.0);

8. Create the zone chart. This is a file with each state or country and what zone its in on each line. Here is a few lines from an example of a Zone Chart for Countries:

   "Country", Zone
   "Afghanistan", 4
   "Albania", 5
   "Algeria", 5
   "Angola", 5
   "Anguilla", 5
   "Antigua & Barbuda", 5
   "Argentina", 5
   "Armenia", 5
   "Ascension & St. Helena", 5
   "Aruba", 5
   "Australia", 1

9. Create the rate chart. If using items table, then each line is the number of items (or a range of items), and a shipping cost for each zone. If using a weight chart, then each line is a weight (or range of weight) followed by a shipping cost for each zone. Commas separate each of these fields. Here is an example of a Rate Chart.

   Items or Weight,Zone1,Zone2,Zone3,Zone4,Zone5
   1,0.00,3.75,4.75,5.75,6.50
   2,0.00,6.50,8.50,10.50,12.00
   3,0.00,6.50,8.50,10.50,12.00
   4,0.00,10.25,13.25,16.25,18.50
   5,0.00,10.25,13.25,16.25,18.50

Download File Function

With CyberCart, you can also sell computer files and have them available for download after the customer completes their order. There are 3 options available, differing on the basis of how secure the process is:

1) **Immediate download.** ($dl_type = 1) The files are made available on completion of the ordering process. This is the least secure (unless you are using Online Analysis for real-time credit card processing), but by far the simpliest method for customer downloading of purchased files. When the customer completes the CyberCart ordering process, a web page listing the files is then displayed and the customer can quickly download the purchased files.
2) **Email confirmation method.** ($dl_type = 2) Upon completion of the ordering process, the customer is sent an additional email message with a download authorization code and the URL of the ccdown.html page. The message is sent automatically before any payment processing (unless you are using Online Analysis for real-time credit card processing), but at least you have a confirmation of the customer's email address.
3) **Merchant Forwarding method.** ($dl_type=3) The merchant is sent an additional email message containing the download authorization code and URL of the ccdown.html page. The merchant can process the payment and then forward the message to the customer.

How it works

Items on the catalog that are files available for downloading are marked with a hidden form tag (see Creating CyberCart Pages). When the customer completes the ordering process, a filelist containing the list of files purchased is created for that customer (xxxxxxxx.files). The files for downloading are stored in a special directory (Files) which should be outside the document root (in the same directory as Data, Logs, and Orders). To retrieve their files, the customer goes to a special CyberCart Download page (ccdown.html) and enters their download authorization code. The download script (ccdown.pl) will check for their filelist and display a link for downloading purchased files. In this way, customers will only have access to files that they have purchased. The customer clicks on the link, and the download begins. The download script also contain a cleanup routine that will check for filelists older than a specified time (variable $expire sets this duration).

How to Set Downloading Up

1. In the merchant configuration file, set the download variables.

```
$dl_type = 2;
#
# URL for the CyberCart Download script
$dl_url = "http://www.name.com/cgi-bin/ccdown.pl";
#
# URL for the Web Page interface where customers can
# enter their download authorization code.
$dl_page = "http://www.name.com/ccdown.html";
# Location of the files to be downloaded. This should be a file
```

```
# outside the root directory.
$dl_files = $home_dir . "\\Files\\";
#
# The merchant_id
$merchant_name = "Demo";
```

2. In the CyberCart HTML pages, add the dlfile hidden variable as describe in Making CyberCart Pages. Set the value to the name of the download file.

3. Create the Files subdirectory in the same directory as Data, Orders, and Logs.

4. Place the files available for download in the Files directory.

5. Change the permissions of the directory and files so the script has access to the files.

6. Place the ccdown.pl script in the cgi directory.

7. Modify the ccdown.pl script.

```
# Merchant root directory
$merchant_data = "c:\\Merchant";

# URL of this script
$dl_url = "http://www.name.com/cgi-bin/ccdown.pl";

# URL of the CyberCart download HTML page
$dl_html = "http://www.name.com/ccdown.html";

# How long to retain the filelist files
$expire_time = 2;

# Make sure to include the mimetype for the type of files
# available for download.
%mimetypes = (
  'pdf' => 'application/pdf',
  'jpg' => 'image/jpeg',
  'doc' => 'application/msword'
);
```

7. Change the permission of ccdown.pl as appropriate.

Administrative Script (getorder.pl)

The getorder.pl script allows you to view orders, delete orders, view order-logging information. To set up the getorder.pl script, follow the instructions given in the installation instructions. Once set up, access the getorder.html page and enter Admin as the Username – you must have already created a .ccusers password file using adduser.pl. The passwords are encrypted using the crypt function, so the file should be in a file outside the root html directory with the permissions set to read only.

Functions

1. **View Orders**. To view merchant orders, the merchant goes to the getorder.html, enters their Username and Password. Then select "List Current Orders" to view orders. You can also delete orders and change the status of the orders.
2. **Summary of Orders.** The merchant can also view a summary of orders, which lists referrer (see Referrer Function for more information), order number, date, and amount of order for each order.
3. **Item Log.** Lists the number and total amount ordered for each item ordered online.
4. **Add Referrer.** Add a referrer id, email address and password (optional).