

## The Progress Electronic Magazine

### In this issue:

<b>Publisher's Statement:</b> .....	2
<b>Coding Article: File Uploading/Downloading With A Progress Database via CHUI, GUI, or WEB:</b> .....	3
The Flow Of Events .....	3
Program Files To Store The File .....	4
Upload1.html .....	4
Upload2.html .....	5
Upload3.html .....	7
StoreArticleFile.p .....	8
StoreFile.p .....	9
Program Files To Retrieve The File .....	11
RetrieveFile.p .....	11
ThrowFile.html .....	12
Database Tables .....	13
OSFile .....	13
OSFileData .....	15
Errata .....	16
<b>Publishing Information:</b> .....	17
<b>Products Available From Amduus:</b> .....	17
<b>Article Submission Information:</b> .....	17
<b>Subscription Information</b> .....	18

*This document is protected by Copyright. Distribution is prohibited.*

*Though intended for users of the software tools provided by Progress Software Corporation, this document is  
NOT a product of Progress Software Corporation.*

© Amduus Information Works, Inc. 2002

**Publisher's Statement:**

This edition focuses on uploading and downloading files from a Progress database. It includes code for both CHUI/GUI/WWW interfaces to your Progress programs. It also can be used for transporting files across computer systems without the need of FTP or NFS, etc. for a pure Progress solution.

I would like to announce that I am looking for resellers for the Portal and Blue Diamond software.

Blue Diamond is an inexpensive means of access to your Progress or DataServer based databases using the E4GL programming language. Blue Diamond includes some routines and templates not available in the Progress Webspeed product that will help you get your customer off and running faster than using Webspeed. It is already in use at companies.

In addition, I am also selling the Portal software. This software can be used to make corporate portals on the intranet, extranet, or internet. It has many features and PDFs are available documenting the benefits of this software.

More information including sales PDFs about both of these products can be found at the [www.amduus.com](http://www.amduus.com) web site or by contacting me directly.

To your success,  
Scott Auge  
Founder, Amduus Information Works, Inc.  
[sauge@amduus.com](mailto:sauge@amduus.com)

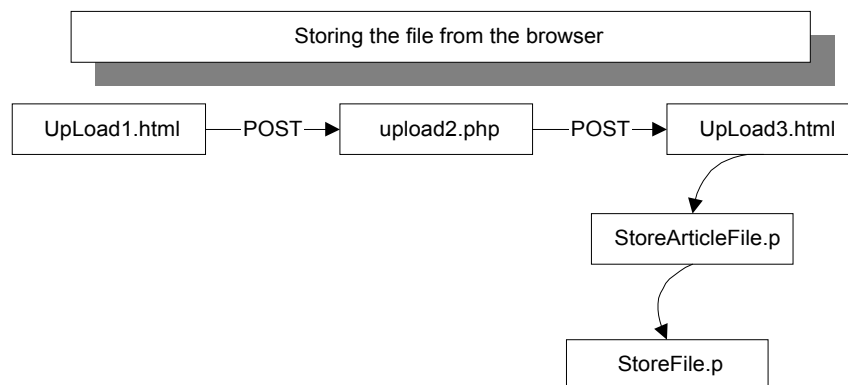
**Coding Article: File Uploading/Downloading With A Progress Database via CHUI, GUI, or WEB.**

*Written by Scott Auge sauge@amduus.com*

To use the code in CHUI and GUI interfaces, the reader will wish to focus on StoreFile.p and RetrieveFile.p versus the rest of the article. One could practically cut and paste them from this text and be functional (provided the Database tables exist), so no great detail will be made in explaining them.

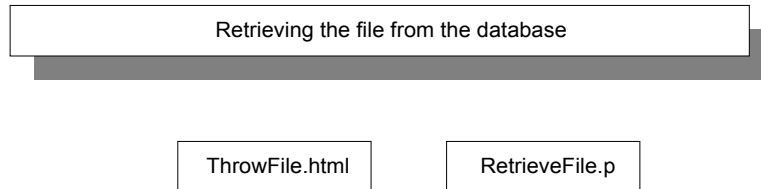
***The Flow Of Events***

One of the ways to store a file from the web is profiled below. Progress Webspeed at the time of this writing does not support binary file uploads. To work around this we use PHP which is available on a wide variety of OS's and Web Servers.



We set up the input form in UpLoad1.html. Inside this page are hidden fields that will be passed along to UpLoad3.html via upload2.php so we can better define where the file falls into place with other data. This information is called the ArticleID – a piece of data used in Amduus Information Works, Inc. portal software. Upload2.php handles the moving of the file from the browser onto the operating system file system and then prompts the user to continue on to UpLoad3.html.

UpLoad3.html will use Progress 4GL to load that file into the database and return a code on the success or failure of the storage.



A simple Progress 4GL can throw out binary information stored in the database to the file system of the operating system. A little more complicated file can be used to throw the file out to the browser.

### ***Program Files To Store The File***

The file receives a NVP (Name/Value Pair) identifying the article id that should be associated with the file. The program then re-renders that article id into a hidden field in the form used to pass in the file to the PHP program which has the correct architecture to receive files in forms.

#### *Upload1.html*

```

<!--WSS
DEF VAR RCSVersion AS CHARACTER INIT "$Header:
/home/appl/opensrc/portal/src/Adm
in/RCS/UpLoad1.html,v 1.1 2002/06/30 05:07:02 sauge Exp sauge $" NO-UNDO.

/* This is nasty, but it works until I get the conf file stuff in here. */

&GLOBAL-DEFINE UPLOADURL "http://amduus2/php/upload2.php"

/* Be sure to check php.ini has the proper setting for this - usually 2MB */

&GLOBAL-DEFINE MAX_FILE_SIZE 2000000

{ShowArticleFiles.i}

DEF VAR cArticleID AS CHARACTER NO-UNDO.

ASSIGN cArticleID = GET-VALUE("ArticleID").

-->

<HTML>
<HEAD>
<TITLE>File Upload</TITLE>
<style type="text/css">

```

```

<!--
body { font-family: Arial, Helvetica, sans-serif; font-size: 10pt}
-->
</style>
</HEAD>

<BODY>
<center>
<table>
<FORM ENCTYPE="multipart/form-data" ACTION="\{&UPLOADURL}\`" METHOD="POST">
<INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="\{&MAX_FILE_SIZE}\`">
<input type="hidden" name="ArticleID" value="\cArticleID\`">

<tr>
<td><b>File Upload</b></td>
<td></td>
</tr>
<tr>
<td>File: </td>
<td><input type="File" name="userfile" size="30" maxlength="255">
</td>
</tr>
<tr>
<td colspan="2" align="CENTER">
<INPUT TYPE="submit" VALUE="Upload">
</td>
</tr>
</FORM>
</table>
</center>
</BODY>
</HTML>

```

Note that there is a MAX\_FILE\_SIZE. This is used by the PHP system to prevent huge files from being uploaded. There is also a value in php.ini that needs to be adjusted to match or be greater than the MAX\_FILE\_SIZE.

Make note of the <form> ENCTYPE – this is very important for sending files along to the web application.

### Upload2.html

This file is presented for completeness of the article, however the E-Zine does not focus on PHP programming. In short, the page receives the file from the browser. When PHP receives a file, it stores it in a temporary name in the directory specified in the php.ini file. This code will take that temporary name and name it as it was in the browser and move it to a location that

---

StoreArticleFile.p found in Upload3.html can find it. It will then present a link for the user to click on to move the browser to Upload3.html with some additional information to help that program further process the file into the database and link it to existing data in the application.

```
<?php
// $Header$

// This file will receive the file from the user's browser as specified by
// Upload1.html in the Portal System (portal/src/Upload1.html)
// Once we got the file on the web server, we need to get the file into
// the database. This is done by calling Upload3.html found in (portal/
// src/Upload3.html) which calls the appropriate Progress 4GL procedure

// BE SURE TO CHECK THE ACTION IN THE FORM BELOW TO GO TO THE APPROPRIATE PLACE

function do_upload($filename,$newname) {
    $file = basename($filename);
    $tmp_upload_path = "/tmp/";
    $new_file_name = "/tmp/". $newname;
    if (!copy($tmp_upload_path.$file, $new_file_name)) echo "failed to copy
file<b
r>\n";
    return;
}
?>

<HTML>

<HEAD>
<TITLE>File Upload</TITLE>
<style type="text/css">
<!--
body { font-family: Arial, Helvetica, sans-serif; font-size: 10pt}
-->
</style>
</HEAD>

<BODY BGCOLOR="#FFFFFF" TEXT="#000000" LINK="#0000FF" VLINK="#800080">
<?php
// userfile is the temporary name of the file in the directory specified in
// php.ini. userfile_name is the name of the file as it is on the user's
// computer and this is the file name that we are interested in.

do_upload($userfile,$userfile_name);
?>
<TABLE>
    <TR>
```

---

```

        <TD><b>upload report</b></TD><TD></TD>
    </TR>
    <TR>
        <TD>file name:</TD><TD><?php echo $userfile_name; ?></TD>
    </TR>
    <TR>
        <TD>target file name:</TD><TD><?php echo $userfile_name; ?></TD>
    </TR>
    <TR>
        <TD>file size:</TD><TD><?php echo $userfile_size; ?></TD>
    </TR>
    <TR>
        <TD>file type:</TD><TD><?php echo $userfile_type; ?></TD>
    </TR>
</TABLE>
<form method = "POST" action="http://amduus2/online/portal/Admin/UpLoad3.html">
<table>
<tr>
<td>
<input type="hidden" name="ArticleID" value="<?php echo $ArticleID; ?>">
<input type="hidden" name="FileName" value="<?php echo $userfile_name; ?>">
<input type="hidden" name="FileType" value="<?php echo $userfile_type; ?>">
<input type="submit" name="submit" value="Continue">
</td>
</tr>
</table>
</form>

</BODY>
</HTML>

```

Normal that the upload2.php file passes along to the UpLoad3.html program some information that the Progress routines will need to correctly insert the file into the database.

### Upload3.html

The Upload3.html page is referenced from the Upload2.php page. It's purpose is to take the file from the location that PHP routines deposited it and place it into the database. The ArticleID is used to help identify how to relate the file to the rest of the data in the system. The Content-type and File Name are used to find the file on the operating system that PHP is running on. Once complete, it will inform the user about success or failure.

```

<!--WSS
DEF VAR RCSVersion AS CHARACTER INIT "$Header:
/home/appl/opensrc/portal/src/Adm
in/RCS/UpLoad3.html,v 1.1 2002/06/30 05:07:02 sauge Exp sauge $" NO-UNDO.

```

```

DEF VAR cArticleID    AS CHARACTER NO-UNDO.
DEF VAR cFileName     AS CHARACTER NO-UNDO.
DEF VAR cContentType  AS CHARACTER NO-UNDO.
DEF VAR cError        AS CHARACTER NO-UNDO.

```

```
{ShowArticleFiles.i}
```

```
ASSIGN
```

```

cArticleID = GET-VALUE("ArticleID")
cFileName  = GET-VALUE("FileName")
cContentType = GET-VALUE("FileType").
-->

```

```

<html>
<HEAD>
<TITLE>File Upload</TITLE>
<style type="text/css">
<!--
body { font-family: Arial, Helvetica, sans-serif; font-size: 10pt}
-->
</style>
</HEAD>

<BODY BGCOLOR="#FFFFFF" TEXT="#000000" LINK="#0000FF" VLINK="#800080">
Loading File Into Database <BR>And Associating to Article.<br>
<!--WSS
RUN file/StoreArticleFile.p (
INPUT cArticleID,
INPUT cFileName,
INPUT "/tmp/",
INPUT cContentType,
OUTPUT cError
).
-->

```

```
File Loaded With Result: `cError`
```

```

<center><a href="dummy" onclick="window.close(); return
false;">Close</a></cente
r>
</body>
</html>

```

*StoreArticleFile.p*



This routine calls the StoreFile.p routine. It is here to give you some ideas of how you may wish to associate a file to some form of data. This code is from the Web Content Management System available from Amduus Information Works, Inc. and associates a file to a web page (called an Article in the system.)

```
DEF VAR RCSVersion AS CHARACTER INIT "$Header:
/home/appl/opensrc/portal/src/fil
e/RCS/StoreArticleFile.p,v 1.1 2002/06/30 00:49:29 sauge Exp $" NO-UNDO.
```

```
DEF INPUT PARAMETER cArticleID    AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cFileName     AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cDirectory    AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cContentType  AS CHARACTER NO-UNDO.
DEF OUTPUT PARAMETER cError       AS CHARACTER NO-UNDO.
```

```
DEF VAR cOSFileID    AS CHARACTER NO-UNDO.
```

```
{Error.i}
```

```
RUN file/StoreFile.p (
INPUT cFileName,
INPUT cDirectory,
INPUT cContentType,
OUTPUT cOSFileID,
OUTPUT cError
).
```

```
IF cError <> {&NOERROR} THEN RETURN.
```

```
CREATE XArticleOSFile.
```

```
ASSIGN XArticleOSFile.ArticleID = cArticleID
      XArticleOSFile.OSFileID = cOSFileID.
```

```
ASSIGN cError = {&NOERROR}.
```

### StoreFile.p

This routine actually reads in a file from the operating system file system and stores it into the database. It's counterpart is called RetrieveFile.p and is listed later on. It is important to remember that it uses the RAW data type to store the file information in. Since you have control over the size of the data you are reading in, you may wish to make it the same as your database record block size to more densely pack the table.

```
DEF VAR RCSVersion AS CHARACTER INIT "$Header:
/home/appl/opensrc/portal/src/fil
```

```
e/RCS/StoreFile.p,v 1.2 2002/07/01 02:35:55 sauge Exp sauge $" NO-UNDO.
```

```
DEF INPUT PARAMETER cFileName      AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cDirectory     AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cContentType AS CHARACTER NO-UNDO.
DEF OUTPUT PARAMETER cOSFileID     AS CHARACTER NO-UNDO.
DEF OUTPUT PARAMETER cError        AS CHARACTER NO-UNDO.
```

```
DEF VAR rData AS RAW NO-UNDO.
DEF VAR iSequence AS INTEGER NO-UNDO INITIAL 1.
```

```
{Error.i}
{LastStringCharacter.i}
{MakeID3.i}
```

```
/* Best to make this the size of a database block. */
```

```
ASSIGN LENGTH(rData) = 16384.
```

```
/* Make sure the directory has a delimiter on it. */
```

```
IF LastStringCharacter(cDirectory) <> "/" THEN
  ASSIGN cDirectory = cDirectory + "/".
```

```
/* Make the header, then load in the data into multiple records if */
/* we hit past the max record size.                                */
```

```
CREATE OSFile.
```

```
ASSIGN OSFile.OSFileID = MakeID3(10).
```

```
ASSIGN
OSFile.FileName = cFileName
OSFile.OSFileType = cContentType
OSFile.CreateDate = TODAY
OSFile.CreateTime = TIME.
```

```
/* Load into the Data table for a OSFile. */
```

```
INPUT FROM VALUE(cDirectory + cFileName) BINARY NO-MAP NO-CONVERT.
```

```
REPEAT:
```

```
  IMPORT UNFORMATTED rData.
```

```
  CREATE OSFileData.
```

```
  ASSIGN OSFileData.OSFileID = OSFile.OSFileID.
```

```

    ASSIGN OSFileData.Sequence = iSequence
           OSFileData.Data      = rData

           iSequence            = iSequence + 1.

END. /* REPEAT */

INPUT CLOSE.

OS-DELETE VALUE(cDirectory + cFileName).

ASSIGN cOSFileID = OSFile.OSFileID
       cError = {&NOERROR}.

```

### ***Program Files To Retrieve The File***

#### *RetrieveFile.p*

This is a simple routine. Given a file identification (the code this is from, uses has a foreign key as the identifier for version controls in the software), output the file into cFileName in directory cDirectory.

It will place a file from the database as stored by the StoreFile.p program back to the operating system file system.

This is a good way to send files over a network if you wish to use a pure progress solution (no FTP or NFS). It can be used with GUI and CHUI interfaces.

```

DEF VAR RCSVersion AS CHARACTER INIT "$Header:
/home/appl/opensrc/portal/src/file/RCS/RetrieveFile.p,v 1.1 2002/06/30 00:49:29
saugue Exp $" NO-UNDO.

DEF INPUT PARAMETER cOSFileID      AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cFileName      AS CHARACTER NO-UNDO.
DEF INPUT PARAMETER cDirectory     AS CHARACTER NO-UNDO.
DEF OUTPUT PARAMETER cError        AS CHARACTER NO-UNDO.

{Error.i}
{LastStringCharacter.i}

/* Make sure the directory has the correct delimiter at the end. */

IF LastStringCharacter(cDirectory) <> "/" THEN
  ASSIGN cDirectory = cDirectory + "/".

```

```
FIND OSFile NO-LOCK
WHERE OSFile.OSFileID = cOSFileID
NO-ERROR.

IF NOT AVAILABLE OSFile THEN DO:

    ASSIGN cError = {&NORECORD}.
    RETURN.

END.

/* Determine if we are using the stored file name or a new one */

IF cFileName = ? THEN
    ASSIGN cFileName = OSFile.FileName.

/* Dump the file out */

OUTPUT TO VALUE(cDirectory + cFileName) NO-MAP NO-CONVERT BINARY.

FOR EACH OSFileData OF OSFile NO-LOCK
BY Sequence:
    PUT CONTROL OSFileData.Data.
END.

OUTPUT CLOSE.

ASSIGN cError = {&NOERROR}.
```

### ThrowFile.html

The ThrowFile.html program will send the file out to a browser instead of to the file system of the operating system. One should determine the content type before storing the file – as is done in the OSFileType field of the OSFile table. A list of content types can be found at Amduus Information Works, Inc. in the search area. Enter MIME and click GO. It should bring up pages stored at the site with that information.

```
DEF VAR cOSFileID      AS CHARACTER NO-UNDO.
DEF VAR rChunkOfFile   AS RAW NO-UNDO.

PROCEDURE OUTPUT-HEADERS:

    ASSIGN cOSFileID = GET-VALUE("OSFileID").

    FIND OSFile NO-LOCK
```

---

```

WHERE OSFile.OSFileID = cOSFileID
NO-ERROR.

Output-Content-Type (OSFile.OSFileType).

END. /* PROCEDURE OUTPUT-HEADERS() */

FOR EACH OSFileData OF OSFile
BY Sequence:

/* Webspeed uses a different stream name than Blue Diamond does, and */
/* to do this, we need to access the stream name directly. So this */
/* program works in both Webspeed and Blue Diamond, we need to de- */
/* termine if WEBSTREAM is defined and compile appropriately. */

&IF DEFINED(WEBSTREAM) > 0 &THEN
PUT STREAM WebStream CONTROL OSFileData.Data.
&ELSE
PUT STREAM StdOut CONTROL OSFileData.Data.
&ENDIF

END.

```

## Database Tables

### OSFile

07/09/02 18:50:23 PROGRESS Report  
Database: amduus (PROGRESS)

```

=====
===== Table: OSFile =====

```

Table Flags: "f" = frozen, "s" = a SQL table

Table Name	Dump Name	Table Flags	Field Count	Index Count	Table Label
OSFile	osfile		7	2	OSFile

Storage Area: Schema Area

===== FIELD SUMMARY =====  
 ===== Table: OSFile =====

Flags: <c>ase sensitive, <i>ndex component, <m>andatory, <v>iew component

Order	Field Name	Data Type	Flags	Format	Initial
10	FileName	char	i	x(8)	
40	OSFileType	char		x(8)	
50	OSFileID	char	i	x(8)	
60	CreateDate	date	i	99/99/99	?
70	CreateTime	inte	i	->, >>, >>9	0

Field Name	Label	Column Label
FileName	FileName	FileName
OSFileType	OSFileType	OSFileType
OSFileID	OSFileID	OSFileID
CreateDate	CreateDate	CreateDate
CreateTime	CreateTime	CreateTime

===== INDEX SUMMARY =====  
 ===== Table: OSFile =====

Flags: <p>primary, <u>nique, <w>ord, <a>bbreviated, <i>nactive, + asc, - desc

Flags	Index Name	Cnt	Field Name
	key1	3	+ FileName + CreateDate + CreateTime
pu	pukey2	1	+ OSFileID

\*\* Index Name: key1  
 Storage Area: Schema Area  
 \*\* Index Name: pukey2  
 Storage Area: Schema Area

===== FIELD DETAILS =====  
 ===== Table: OSFile =====

OSFileData

07/09/02 18:50:50            PROGRESS Report  
 Database: amduus (PROGRESS)

=====

===== Table: OSFileData =====

Table Flags: "f" = frozen, "s" = a SQL table

Table Name	Dump Name	Table Field Flags	Index Count	Table Count	Table Label
OSFileData	osfileda		4	1	OSFileData

Storage Area: Schema Area

===== FIELD SUMMARY =====

===== Table: OSFileData =====

Flags: <c>ase sensitive, <i>ndex component, <m>andatory, <v>iew component

Order	Field Name	Data Type	Flags	Format	Initial
20	Sequence	inte	i	->, >>>, >>9	0
50	OSFileID	char	i	x(8)	
60	Data	raw		x(8)	

Field Name	Label	Column Label
Sequence	Sequence	Sequence
OSFileID	OSFileID	OSFileID
Data	Data	Data

===== INDEX SUMMARY =====

===== Table: OSFileData =====

Flags: <p>primary, <u>nique, <w>ord, <a>bbreviated, <i>nactive, + asc, - desc

Flags	Index Name	Cnt	Field Name
-----			
p	pukey1	2	+ OSFileID + Sequence

\*\* Index Name: pukey1  
Storage Area: Schema Area

===== FIELD DETAILS =====  
===== Table: OSFileData =====

### ***Errata***

This software was developed on Red Hat Linux 6.2 with Progress Version 9.1C. Webspeed 3.1C and Blue Diamond Build 2002.149.03.40 were also used in the development of the code. The code comes from the Portal system sold and shared by Amduus Information Works, Inc.

*About the author: Scott Auge is the founder of Amduus Information Works, Inc. He has been programming in the Progress environment since 1994. His works have included E-Business initiatives and focuses on web applications on UNIX platforms.*  
[sauge@amduus.com](mailto:sauge@amduus.com)



**Publishing Information:**

Scott Auge publishes this document. I can be reached at [sauge@amduus.com](mailto:sauge@amduus.com).

Amduus Information Works, Inc. assists in the publication of this document:

Amduus Information Works, Inc.  
1818 Briarwood  
Flint, MI 48507  
<http://www.amduus.com>

**Products Available From Amduus:**

**Blue Diamond** – Use your E4GL code in this Webspeed alternative for the UNIX/Linux operating system (some compatibility with Windows)

**Service Express** – Web based work order/ticket system with work-flow capabilities.

**Denkh** – Create PDF documents/reports on your Linux/UNIX computer! Headers, Footers, bar charts, as well as tabular data results. Useful for a web based reporting system as well as CHUI and GUI.

**Portal** – Web content management system. Add web pages easily with linking and searching automatically accomplished.

**Mail** – Used to distribute this very E-Zine to 800+ readers. Allow your customers to know what you can accomplish for them with newsletters and ezines they can sign up for via the web.

We also lease these applications as well perform custom programming and Progress license sales.

**Article Submission Information:**

Please submit your article in Microsoft Word format or as text. Please include a little bit about yourself for the About the Author paragraph.

Looking for technical articles, *marketing Progress* articles, articles about books relevant to programming/software industry, white papers, etc.

### **Subscription Information**

Send \$50.00 for twelve issues to:

Amduus Information Works, Inc.  
1818 Briarwood  
Flint, MI 48507

Your Name: \_\_\_\_\_

Your Email Address: \_\_\_\_\_

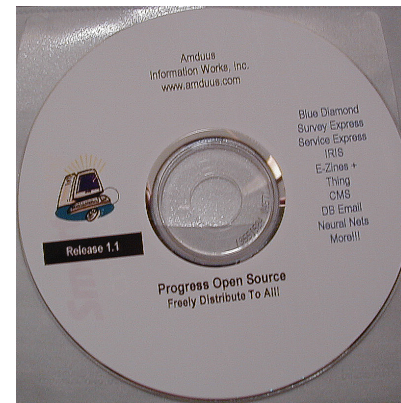
Your Company Name: \_\_\_\_\_

Order Form for Progress Open Source CD-ROM  
COUPON 001A

This is an offer for the CD-ROM at lower than list savings!

Mail this form to:  
**Amduus Information Works, Inc.**  
**1818 Briarwood**  
**Flint, MI 48507**

Please send \_\_\_\_\_ copies of the Open Source CD-ROM at  
\$25.00 per disk to:



Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City \_\_\_\_\_  
State \_\_\_\_\_  
Zip \_\_\_\_\_

Please make your checks/money orders out to: [Amduus Information Works, Inc.](#) Cash works too!  
This offer only valid in the United States of America.

The CD-ROM includes (all source code included):

- Blue Diamond/IRIS – Webspeed alternatives
- Survey Express – easily create text templates of surveys and then have the program generate the web pages automatically
- Service Express – Web based Help Desk.
- The Progress E-Zines, books on learning to program in Webspeed (PDF/Word/HTML)
- THING – simple tool to manipulate database records with
- CMS – a web content management system
- DB Email – Use pop3 to download emails into a Progress database
- Neural Networks – experiments in spam recognition and text message classification
- GenPDF – create PDF file reports for Webspeed/UNIX CHUI!
- More!