Back Up Your Projects

Here's a tool that lets you back up all the components of a project easily—so you can take it with you.

By Paul Ferretti

PRODUCT

Visual FoxPro

Recently, a project I was working on required some extra work to be done at home. I needed a simple and thorough way to back up my work files and restore them on my home computer. The backup application was originally developed in Visual FoxPro, and has since been modified to work in the 2.x environment as well. (You'll find the Visual FoxPro version on this issue's *Companion Resource Disk*.) In either case, the application requires PKZip to handle the file compression, and PKUnzip to handle the file decompression for reporting purposes.

What makes this application so useful is that it not only backs up all files referenced in the PJX file, it also finds all tables, memo files, and indexes associated with a database container file.

The application consists of one form (.SCX), one report (.FRX), the FoxTools.FLL library, and one class library that was created for this application. It uses one table (PJXDEFA.DBF) which holds the location of PKZIP, PKUNZIP, the last .PJX selected, and the path last used as the backup destination.

To locate PKZip and PKUnzip, the application first checks the table values (cZipPath and cUnZipPath) to see whether they're valid. If either isn't found, the application prompts you using LOCFILE(). After you locate the files, you never have to locate them again—unless you move them. If PKZip and PKUnzip are in the same directory, you only have to locate PKZIP.

If, for some reason, PKZip or PKUnzip isn't available, you can still run the application, but you won't be able to execute the function associated with the missing file (the Backup function for PKZip and the Print function for PKUnzip).

Continued

Paul Ferretti is director of software development for Coe Management Services International, located in Fairfax, Virginia. He's spent the past eight years designing and developing Xbase systems for the government and private sector. (703)275-6076. **71554.1007@compuserve.com, ferretti@erols.com**.

The application warns you about this at startup and doesn't let you continue without at least one of these two files.

The report is called PJXBKUP.FRX (figure 1). The report provides a listing of file names and paths included in the backup file, along with the location, size, and date/time stamp for each file.

Figure 2 shows the main form (BKUPFORM.SCX), which consists of a text box to enter the full path and name of the project file to be backed up, and a text box to enter only the path of the destination.

Project Bacup Log

PJX Backed Up: C:\VFP\MYPROJ.PJX

Backed Up To: C:\BACKUP\MYPR1025.ZIP

Path Name	File Name	Type	Size	Date	Time
C:\VFP\MYPROJ\BMPS					
	LOGO.BMP	BMP	7018	07-20-95	22:20
	ADD.BMP	BMP	5019	07-20-95	23:16
C:\VFP\MYPROJ\DATA					
	MYPROJ.DBC	DBC	87013	08-27-95	22:13
	MYPROJ.DCT	DCT	60160	08-27-95	22:13
	MYPROJ.DCX	DCX	12800	08-27-95	22:13
C:\VFP\MYPROJ\FORMS					
	FORM1.SCT	SCT	2955	08-27-95	21:51
	FORM1.SCX	SCX	2015	08-27-95	21:51
	FORM2.SCT	SCT	4194	08-27-95	21:51
	FORM2.SCX	SCX	2886	08-27-95	21:51
	FORM3.SCT	SCT	5231	08-27-95	21:51
	FORM3.SCX	SCX	3532	08-27-95	21:51

Page 1 10/25/95 10:00

Figure 1: **PROJECT BACKUP REPORT**—The report shows the files backed up, along with type, size, date and time.

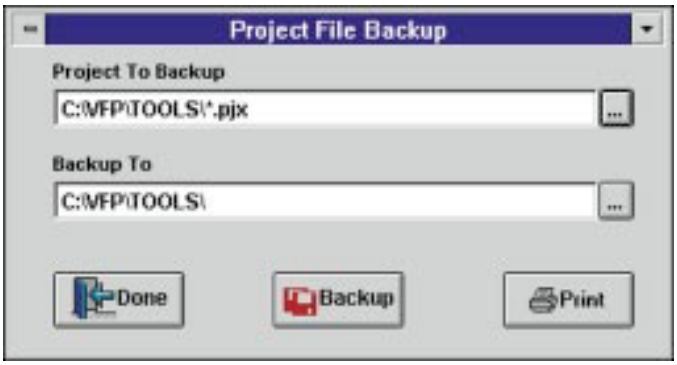

Figure 2: **APPLICATION MAIN SCREEN**—Specify the project to be backed up and the destination.

Directory buttons are provided so the user can select from a directory dialog (which is simply a call to the GETDIR() function), if he or she can't remember the exact path or name of the project file.

The code for the Backup and Print commands is in two custom methods of the form. The method for creating the backup is MakeBackup, and the method for printing the report is called PrntLog.

The MakeBackUp method creates a .ZIP file with a name consisting of the first four characters of the .PJX file, plus the current month and day.

For example, the project file MYPROJ.PJX backed up on 12/26/95 would have a backup file named MYPR1226.ZIP, as follows:

```
*********************************************************
* Program.....:  MakeBackUp
* Author......:  PjF
* Date........:  10/26/95,11:00:32
* Description..:  This method creates a zip file based
*              :  on a PJX file.
* Parameters...:  tcPjxFile - The name and path of the
*              :  project file to backup.
*              :  tcBkUpPath - The path to the location of
*              :  the backup file.
*********************************************************

PARAMETER tcPjxFile, tcBkUpPath

IF !EMPTY(tcPjxFile)
  Thisform.Thermometer()

  */
  */ Save where we are.
  */
  cOldDir = SYS(5)+CURDIR()

  */
  */ Go to backup directory.
  */
  cNewDir = JustPath(tcPjxFile)
  SET DEFA TO (cNewDir)
  cThisDir = SYS(5)+CURDIR()

  */
  */Open project file as a table.
  */
  USE (tcPjxFile) ALIAS CurPjx
  nItem = 0

  */
  */ Gather some information about the file, and
  */ create the ZIP file name.
  */
  cFname = JustStem(tcPjxFile)
  cDateStr = SUBSTR(DTOS(DATE()),5,4)
  cBkUpFile = tcBkUpPath+LEFT(cFname,4)+cDateStr+'.zip'

  oTherm.updatebar(5)

  IF !FILE('fname.tmp')
    nHndl = FCREATE('fname.tmp')
  ELSE
    nHndl = FOPEN('fname.tmp')
  ENDIF

  IF FILE('tempzip.zip')
    DELETE FILE tempzip.zip
  ENDIF

  oTherm.updatebar(10)

  nItem = 0

  SCAN
    cTxt = MLINE(name,1)
    DO CASE
      CASE type = 'd'
```

Continued

```
        nItem = nItem + 1
        DIMENSION aDbc[nItem]
        aDbc[nItem] = cTxt
      CASE type = 'H'
        cTxt = JustFname(cTxt)
    ENDCASE
    cTxt = JustPath(FULLPATH(cTxt))+ ;
      IIF(!EMPTY(JustPath(FULLPATH(cTxt))),'\','')+ ;
      JustStem(FULLPATH(cTxt))+'.*'
    =FPUTS(nHndl,cTxt)
  ENDSCAN

  oTherm.updatebar(25)

  USE IN CurPjx

  FOR nCurItem = 1 TO nItem
    USE (aDbc[nCurItem]) IN 0 ALIAS tempDbc
    cPathTxt = aDbc[nCurItem]
    SELECT DISTINCT objectname ;
      FROM tempDbc ;
      WHERE objecttype = 'Table' OR objecttype = 'Index' ;
    INTO CURSOR tempTables

    oTherm.updatebar(35)

    SCAN
      cTxt = ALLTRIM(objectname)
      cTxt = JustPath(cPathTxt)+ ;
             IIF(!EMPTY(JustPath(cPathTxt)),'\','') ;
             +cTxt+'.*'
      IF OCCURS('..',cTxt) = 0
        cTxt = cThisDir+ctxt
      ELSE
        cTxt = FULLPATH(cTxt)
      ENDIF
      =FPUTS(nHndl,cTxt)
    ENDSCAN
    oTherm.updatebar(45)
    USE IN TempDbc
    USE IN TempTables
  NEXT

  =FCLOSE(nHndl)

  oTherm.updatebar(50)

  cZipCommand = ALLTRIM(pjxdefa.cZipPath)
  ! &cZipCommand tempzip @fname.tmp -P

  oTherm.updatebar(75)
  IF FILE('tempzip.zip')
    COPY FILE tempzip.zip TO &cBkUpFile
    oTherm.updatebar(100)
  ELSE
    =MESSAGEBOX(cNoZipMsg,16,'Could Not Create File')
  ENDIF
  RELEASE oTherm
  SET DEFA TO (cOldDir)
ENDIF
```

The PrntLog method of the form prints a log of the files contained in your backup file.

When the user clicks on the **Print** button, the PrntLog method is executed. GETFILE() is called to let the user specify the backup file on which to report.

The code for obtaining the information from the .ZIP file and formatting for reporting is a bit ugly, but it seems to do the trick (I'm looking for a more elegant solution, and welcome any suggestions):

```
************************************************************
* Program......:  PrntLog
* Author.......:  PjF
* Date.........:  2/2/96,10:23:57
* Description..:  Creates a report from information
*                 obtained from a ZIP file.
************************************************************
*/
*/ Keep track of where we are when we start the print
*/ operation.
*/
cOldDir = SYS(5)+CURDIR()
cOldFilePath = Thisform.cFilePath.Value
cOldBkupPath = Thisform.cBkUpPath.Value

*/
*/ What ZIP file do we want.
*/
cZipFile = GETFILE('ZIP',"ZIP File","Select")

IF !EMPTY(cZipFile)
  Thisform.cBkUpPath.Value = cZipFile
  Thisform.Thermometer()

  oTherm.updatebar(10)
  cNewDir = JustPath(cZipFile)
  SET DEFA TO (cNewDir)

  */
  */ -vb displays file info in "Brief"
  */ format.
  */
  cUnZipCommand = ALLTRIM(pjxdefa.cUnZipPath)
  ! &cUnZipCommand &cZipFile -vb > FileList.tmp

  oTherm.updatebar(20)
  CREATE CURSOR TempDbf ;
    (FileSize C(10), ;
     CompType C(10), ;
     CompSize C(10), ;
     CompPerc C(10), ;
     FileDate C(8) , ;
     FileTime C(8) , ;
     FullName C(70))

  oTherm.updatebar(30)
  nHndl1 = FOPEN("filelist.tmp",12)
  cLine = ''

  oTherm.updatebar(40)

  */
  */ Move the file pointer until you reach the first
  */ line that contains information about the zipped
  */ files.
  */
  DO WHILE !("---" $ cLine )
    cLine = FGETS(nHndl1)
  ENDDO
  cLine = ''
  oTherm.updatebar(50)

  nHndl2 = FCREATE("dbflist.tmp")

  */
  */ Process the file and format the info to be comma
  */ delimited,until no more relevant information is
  */ left to deal with. This is ugly, but at the
  */ moment I can't think of a better way to deal
  */ with it.
  */
  DO WHILE !("---" $ cLine )
    cLine = FGETS(nHndl1)
    IF !("---" $ cLine )
      cLine = ALLTRIM(cLine)
      cLine = STRTRAN(cLine," ",",")
      cLine = STRTRAN(cLine,",,",",")
      cLine = STRTRAN(cLine,"/","\")
      =FPUTS(nHndl2,cLine)
    ENDIF
  ENDDO
  oTherm.updatebar(60)

  =FCLOSE(nHndl1)
  =FCLOSE(nHndl2)
  oTherm.updatebar(70)

  SELECT TempDbf

  APPEND FROM dbflist.tmp TYPE DELIMITED WITH ,
  oTherm.updatebar(80)
  */
  */ Format the full file name with drive,path and
  */ name info. This makes it easier to break out on the
```

```
*/ report.
*/
REPLACE ALL fullname WITH ;
   JustDrive(FULLPATH(fullname))+"\"+ ;
   JustPath(fullname)+"\"+JustFname(fullname)

*/
*/ Obtain the full project name for the report.
*/
LOCATE FOR JustExt(fullname) = "PJX"
IF FOUND()
  Thisform.cFilePath.Value = UPPER(JustPath(fullname))+ ;
                             "\"+JustFname(fullname)
ELSE
  Thisform.cFilePath.Value = "Unknown"
ENDIF

oTherm.updatebar(90)

*/
*/ Another quick select to order the data for
*/ the report.
*/
SELECT filesize,filedate,filetime,fullname ;
  FROM tempdbf ;
INTO CURSOR tmpfile ORDER BY fullname

oTherm.updatebar(100)
RELEASE oTherm

IF MESSAGEBOX(cPrintMsg,36,'Print File') = 6
  REPORT FORM pjxbkup.frx PREVIEW
ENDIF
REPORT FORM pjxbkup.frx TO PRINT PROMPT NOCONSOLE


*/
  */ Clean up TMP files. There's no need to leave
  */ a mess.
  */
  DELETE FILE filelist.tmp
  DELETE FILE dbflist.tmp

  */
  */Reset previous backup paths.
  */
  SET DEFA TO (cOldDir)
  Thisform.cFilePath.Value = cOldFilePath
  Thisform.cBkUpPath.Value = cOldBkupPath

ELSE
  =MessageBox("A ZIP File Was Not Selected. "+;
              "Process Canceled.", 48,"Nothing To Do")
ENDIF
```

To keep the user informed as to the status of the backup and subsequent printing of the log, I use a thermometer. The thermometer class is an adaptation of one provided by Rick Strahl of West Wind Technologies. (See the May 1996 issue of *FoxPro Advisor* for the article about Rick's thermometer class ("Show Your Progress With a Thermometer").

If you develop large applications with files located all over the LAN, and you need to work away from the office, this little application can make your life a whole lot easier. I can't tell you how many times in the past I've zipped up files to take home, only to find out once I get there that I'm missing that one obscure file from the LAN. With this little tool, I no longer have that worry. ■