Barcode Recognition Server

Addendum to halFILE™ Document Manager

Version 3.0
January 13, 2010

Doc # hfw0014_30

Table of Contents

Barcode Recognition Server: Description of Application

Barcode Recognition Server (Barfile32.exe) is an add-on utility to halFILE. This document pertains to Barcode Recognition Server version 3.0 that supports halFILE versions 3.0 or later. For earlier versions, you should refer to the documentation pertaining to that version. Barcode Recognition Server is used to monitor folders on a network and automatically file and index documents as they appear in the folder. Multiple folders can be set up with each folder referencing a particular halFILE database and basket.

Barfile32.exe is typically used to automatically file documents captured by the following:

1. External scan or capture applications on the same network. You could scan a multi-page TIFF document with a different scan application, save the document to the appropriate folder and Barfile32 will file it away.
2. E-mailed documents. If a multi-page TIFF image was attached to an e-mail message, the attachment could be saved to the appropriate folder and Barfile32 will file it.
3. Ftp scanning applications. FTP or Internet scan/capture applications can ftp multi-page TIFF images to the folder and Barfile32 will then file the document.

An example Barfile32.exe application is:

1. Remote users use ftp scanner to scan documents to a host server.
2. Barfile32.exe splits the TIFF image into single page TIFF images. These images are placed into the appropriate basket folder.
3. Barfile32.exe processes the standard halFILE index routine to add the record to the database by (a) reading the bar code and posting the results into the file number field, (b) auto filling the Date_Filed field with today's date, (c) deleting the original TIFF image.
4. Overnight Auto Archive archives the images.

Program Features/Limitations

1. Barfile32.exe expects documents that are placed into the folder to be multi-page TIFF images.
2. Based on Barfile32.ini settings, as well as parameters within halFILE, Barfile32.exe will perform all of the features that the halFILE Index module is capable of, including Lookups, Bar Coding, Field Auto Fill and integration with Title Plant Closing Applications, such as AIM for Windows and SoftPro ProForms.

Barcode Recognition Server Setup

Select a computer to install Barfile32 on. Barfile32.exe runs all the time, displaying a progress bar. The computer selected should be a low-use user machine or a server. If Barfile32 needs to recognize bar codes, then the computer must have a Kofax card or the Kofax Adrenaline software engine installed.

From the Barfile32 installation disk or folder, run Setup.exe to install Barfile32. Select the halFILE program directory for the target folder.

After the application has been installed, you need to set up Barfile32.ini to tell the application where images that need to be filed will be placed. Barfile32.ini is placed in the halFILE program folder. The following describes the Barfile32.ini options. The parts in curly brackets are documentation only and are not part of the INI file.

[BarFile32]	{The main section for Barfile32}
db1=GATHFWPR	{the database sections that Barfile32.exe should check. You
db2=HALHFWBD	can have up to 99 sections db1, db2… db99 sections. Each section referenced should have a corresponding section as you see below on this example. The section name must be a valid halFILE database name – the first 3 characters is a valid application id, characters 4-6 = 'HFW' and characters 7-8 is a valid database id; additional letters can be added to this tag in order to make it unique, example: HALHFWBD_1 would meet the valid halFILE database name and allow for multiple baskets to be handled by changing the _1 to _2, etc)
IntervalWait=15	{number of seconds to wait after each cycle through all the sections, only valid if OnePass=FALSE. If set to zero then Barfile32.exe automatically defaults to 30 seconds}
FieldTop=100	{barfile32 program screen position – top}
FieldLeft=100	{barfile32 program screen position – left}

FieldWidth=5500 {barfile32 program screen position – width}
FieldHeight=100 {barfile32 program screen position – height}
OnePass=TRUE (TRUE indicates that Barfile32.exe makes one pass through each configured database and then stops. FALSE indicates that Barfile32.exe continuing process until interrupted by the F10 key)
Summary=TRUE {if set to TRUE then Barfile32.exe outputs a single record each time it exits, appending the record to 'barfile32.sum' which will be created in the same location as Barfile32.exe. This record contains: start date/time, finish date/time, elapsed minutes, document count, and page count. I will also have a value of 'FORCED' at the end of the record if the time period for processing images exceeded the value placed in 'hfServiceMgr_ProcessingInterval' if the 'hfServiceMgr' tag is set to true. 'Summary' defaults to 'TRUE' if not set.}
hfServiceMgr=FALSE {if set to TRUE then Barfile32.exe will process for the number of seconds set up in the tag 'hfServiceMgr_ProcessingInterval' (see below) and then it will exit setting the tag 'hfServiceMgr_Restart' to the dbX value that it was processing. This will allow the program to start with the same dbX value when it starts up again. This tag is meant to be used when running barfile32.exe as an NT service utilizing the program 'hfServiceMgr.exe'. If not defined, this defaults to FALSE. If 'hfServiceMgr' is set to true, then 'OnePass' should be set to TRUE as well. Setting hfServiceMgr=TRUE will automatically force NTServiceFlag to TRUE inside Barfile32.exe, which will prevent any displays to the screen.}
hfServiceMgr_Restart=0 {this tag is used in conjunction with 'hfServiceMgr' above. If 'hfServiceMgr=TRUE' and this value is a positive number, then Barfile32.exe will start the next processing cycle using this value for the database X (dbX) starting point. This tag is automatically reset to zero once Barfile32.exe terminates normally}
hfServiceMgr_ProcessingInterval=720 {If 'hfServiceMgr=TRUE', then this value is the number of seconds that Barfile32.exe will process before shutting itself down (and setting hfServiceMgr_Restart to the database currently being processed). This setting works in conjunction with the tag 'ProcessingInterval=' set inside hfservicemgr.ini (and used by hfservicemgr.exe). The idea is to set 'hfServiceMgr_ProcessingInterval' to a value less than 5 times the value of 'ProcessingInterval', which is the amount of time hfservicemgr.exe will wait for barfile32.exe to finish one complete pass before terminating itself. Defaults to 720 seconds or 12 minutes of processing.}
LocalWorkFolder=c:\workarea\ if present, this tag should point to an existing folder on the local workstation where barfile32.exe is running. This folder will be used to make a working copy of the incoming multi-page TIFF before it is split into individual pages by splittif.exe. This setting should always be used if the ImageFolder= values are not local to the workstation where barfile32.exe is running, as splittif.exe will take an extra long time splitting the incoming documents into single pages.
ShutDownTime=hh:mm if OnePass=FALSE, then ShutDownTime= can be used to force barfile32.exe to terminate gracefully at a specific time. The time should be specified in 24-hour military time. If barfile32.exe is running and the time defined by ShutDownTime is passed, then barfile32.exe will stop executing and exit. This is meant to be used when not using the halFILE Service Manager and barfile32 is running in unattended continuous mode. Barfile32.exe will create the file 'Barfile32.dow' in the same location as barfile32.exe when it has successfully shut itself down. By checking for the presence of this file, one can determine if barfile32.exe is or is not currently running.
PatchCount=n If >=n barcodes are found then it splits on patchcode. This prevents it from splitting on patch code if a phone bill or other receipt has a barcode. If your coversheet has 3 barcodes then if you set PatchCount=3 it will only split on barcode if 3 or more barcodes are found on page.

[GATHFWPR] {the first database section}
ImageFolder=c:\gathfwpr\ {the folder where multi-page TIFF images are located}
TargetBasket=Import {the basket into which the images will be split. This must be a valid basket for the designated database}
BasketState=2 {the status of each imported image will be stamped with this value when placed in the basket. BasketState=0 for scan complete, =2 for index complete, defaults to 2 if not defined.}

NoIndex=FALSE {NoIndex=TRUE indicates that for this database, images are split and placed into the basket but no indexing is performed}

[HALHFWBD] {a second database section}

Associated=FALSE {if set to TRUE, then all of the images are treated as associated with another Windows application, such as Word, Excel, ZIP, PDF, and they are not split into individual pages}

AssocMask=*.doc {a file mask used to identify the files which will be imported into the database as associated files; only valid if 'Associated=TRUE'}

ImageFolder=c:\halhfwbd\ {the folder where multi-page TIFF images are located}

TargetBasket=Memos {the basket into which the images will be split. This must be a valid basket for the designated database}

DisableDatabase=FALSE {DisableDatabase is set to 'TRUE' if you wish to have this database ignored by the Barfile32 process}

NoIndex=TRUE

RequiredFields=xx;yy {if tag is defined, then database field numbers 'xx' and 'yy' must have a value loaded or the images are moved to the reject folder defined by the next tag. Field numbers must be separated by semi-colons.}

RejectFolder=c:\rejects\ {the folder where images that are rejected for failing the required field tests are placed. This tag works in conjunction with the tag 'RequiredFields=' above}

RejectBasket=BasketName {the basket where images that fail the 'requiredfields' test will be placed. This must be a valid basket for the designated database. The 'RejectFolder' option takes precedence over the 'RejectBasket'. One or the other should be defined but not both.}

RejectImageFolder=c:\reject2\ {the folder where images that cannot be handled by splittif.exe are placed (bad or corrupted images). If not present then the files are left in the original image folder location}

halfile_images=FALSE {if set to TRUE, then images are not imported(and split) but rather are assumed to be already in a halFILE basket awaiting to be indexed using barcodes. If this tag is set to TRUE then the 'RequiredFields=' tag is ignored and not processed. If the tag is not present, then it defaults to FALSE. If set to TRUE, then only barcode recognition is performed}

SplitOnPatchCode=FALSE {if set to TRUE, then the imported images are split into separate documents based upon a barcode. If the tag is not present, then it defaults to FALSE}

RejectOnLookups=FALSE {if set to TRUE, if any database lookup fails during the import, the images are moved to the reject folder defined by tag 'RejectOnLookupsFolder='. If the tag is not present, then it defaults to FALSE}

RejectOnLookupsFolder=c:\reject3\ {if 'RejectOnLookups=TRUE', then this defines the folder where images that fail any database lookups are placed)

RejectOnLookupsDelay=TRUE {if set to TRUE, then images that are rejected because they fail the database lookup will be placed in a folder defined by 'RejectOnLookupsFolder' and YYYYMMDD where 'YYYY' is the 4 digit year, 'MM' is the 2 digit month and 'DD' is the 2 digit day.}

RejectOnLookupsStaleDate=TRUE {if set to TRUE, then additional passes will be made to re-process images that were rejected for failed lookups on earlier days. The number of passes is defined by the option 'StaleDate=x', where 'x' is the number of business days to look back}

StaleDate=3 {if 'RejectOnLookupsDelay=TRUE' and 'RejectOnLookupsStaleDate=TRUE', then this option determines the number of business days that barfile32 will look back, re-processing already rejected images that failed lookup tests in earlier days. Barfile32 will look for rejected images in folders defined by 'RejectOnLookupsFolder'+ YYYYMMDD for the specific date being processed. When processing the folder for the oldest date, any images that fail the lookup again will be moved to the basket defined by the option 'RejectOnLookupBasket'. Once all images for the oldest date folder have been processed and either been processed successfully or have been rejected due to a lookup failure (and moved to a basket), then the folder (now being empty) will be automatically deleted from the system.}

RejectOnLookupsBasket=LOOKUP {the basket where images that have failed 'StaleDate=' number of lookup failures will be finally moved when Barfile32 has processed the oldest date looking back and still the image fails one or more lookup tests. This must be a valid basket for the designated database.}

BlankPageSize=nnnn { nnnn is the size in bytes that any single page compressed must be in order to be included in the imported image. If defined then all pages are tested to see that they meet this minimum page threshold. Define this tag only if you want this test to be performed}

AutoFieldx=Office {Tag 1 of 2 that can be used to auto populate halFILE database fields when importing images and index information. The 'x' is a number starting at 1 and up to the number of fields in the database, the first set of tags would be 'AutoField1=Office' and its matching tag would be 'AutoValue1=Austin'. In this example 'Office' would be the name of a halFILE database field in the database HALHFWBD}

AutoValuex=Austin {Tag 2 of 2 for auto populating fields in a halFILE database. In this example, 'Austin' is the actual value that will be placed in the field defined by the matching 'AutoFieldx' tag. The 'x' represents a number starting at 1 and going to 99. For each 'AutoFieldx' defined there should be a matching 'AutoValuex' defined. There are a number of reserved words that can be used for the 'AutoValuex' tag that will be replaced with the actual value when the auto fill routine is running. These are:
#NOW – mm/dd/yyyy
#NOWDATE – mm/dd/yyyy
#DATETIME – mm/dd/yyyy hh:mm:ss
#USER – the user id that was used to sign on to halFILE
#TIME – hh:mm
#ASSOCFILENAME – the file name of the original associated file, if the tag 'Associated' is set to TRUE}

BarFileExecute=command {if set, then this external program is executed each time a document is indexed. The 'command' should be the exact command line to execute the external program. Barfile32 suspends further execution until the external program finishes executing. The following substitutions will be performed on the command line before it is executed.
%docnum% - halFILE document number
%appl% - halFILE application id – 3 characters
%doctype% - halFILE document type – 2 characters
%basket% - name of target basket
%dbpath% - data path for database at hand
%tempdir% - windows temporary folder name}

{The next section is not required but if defined would affect the logging capability of Barfile32}

[LogOptions]

NoLog=FALSE {if set to TRUE then no logging will occur. If not present then defaults to FALSE}

LogFilePath=C:\logs\ {the folder where the log file will be created, defaults to the windows temporary folder}

DatedLog=FALSE {if set to TRUE then the log file name created will be 'BARFILE32_YYYYMMDD.LOG' where YYYY is the 4 digit year, MM is the 2 digit month and DD is the 2 digit day. If not present then defaults to FALSE}

LogFilePathError=Log File Path does not exist c:\badpath {only present if a non-existent folder name was provided in the 'LogFilePath' tag. In this example the 'c:\badpath' would be replaced by the actual value that was in the 'LogFilePath' tag initially. BarFile32 automatically corrects for this situation by updating the 'LogFilePath=' tag with the Windows temp folder}

DatedLogBackupPath=c:\logs\backup\ {if 'DatedLog=TRUE' is being used, then this tag defines the folder where the 'BARFILE32_YYYYMMDD.LOG' files will be copied each and every time a new log file is created}

LogFileName=name.log {if defined, overrides the default log file name of 'barfile32.log'}

{if DatedLog=FALSE then the following tag can be defined}

OverwriteLog=TRUE {if set to TRUE, then the previous log file is overwritten each time barfile32 runs. If the tag is not present, then it defaults to TRUE}

Starting and Stopping Barfile32

You can start Barfile32.exe by placing it into Windows Startup folder or by manually starting it.

To stop Barfile32.exe, press F10 twice. The next time the program cycles (based on the Interval Wait), it will unload.

Notes

1. The key value 'Scan Station' should be set up in the section [halFILE for Windows] inside halFILE.ini on the machine that will be running barfile32. This key should be set to a unique single character. It will become the leading character in the document number assigned to all imported images.
2. Barfile32.exe requires a 32-bit Data Source to the halFILE database.
3. If Barfile32.exe fails to connect to the database, the DisableDatabase tag in Barfile32.ini is set to TRUE. You must correct the ODBC datasource so the database connection can be successfully made and edit the Barfile32.ini and set the DisableDatabase tag back to FALSE.
4. For halFILE version 2.1, you must set KofaxStation=TRUE in the [halFILE for Windows] section of halfile.ini. halfile.ini is located in the Windows directory on the computer local drive.
5. The AutoFieldx and AutoValuex tags can appear in either the HFGLOBAL.INI or BARFILE32.INI files. By utilizing just the BARFILE32.INI file, one can do auto-filling of specific fields without affecting the operation of the standard halFILE setup.
6. If you are using external lookups and want to post multiple records from the external table to a halFILE database, then you must set 'PostMultipleLookups=TRUE' within the [xxxHFWyy] section of the HFGLOBAL.INI
7. During barcode conversion, if you want to drop leading zeroes on specific barcodes, you must set 'BarCodeRemoveLeadingZeroes' equal to a list of field numbers you wish to drop leading zeroes from. The field numbers must be separated with a semi-colon. This tag must be in HFGLOBAL.INI.
8. Barfile32.exe uses the external program 'splittif.exe' to split multi-page tiff images. Splittif.exe should exist in the same location as barfile32.exe.
9. If the ImageFolder for any database is not local to the workstation where barfile32.exe is running, then be sure to use the 'LocalWorkFolder=' value to point to a folder on the local workstation that can be used as a work area for splittif.exe.
10. If running Barfile32.exe as an NT service, be sure to set 'OnePass=TRUE' in the [Barfile32] section of BARFILE32.INI.
11. If logging is turned on, the following key words are used inside the log file to quickly identify a variety of different points of interest:
    a. *NOTE* - items of possible interest, including when items get automatically added to the barfile32.ini, the processing time exceeds the time set with 'hfServiceMgr_ProcessingInterval'.
    b. *WARNING* - tags an item when something unexpected happens, but not assumed to be fatal to continuing processing.
    c. *FIXED* - if an invalid value is found in the Barfile32.ini and it can be automatically fixed then it is marked with this.
    d. *ERROR* - used to highlight potential problems, such as problems communicating with a database, folders that do not exist, unable to delete a file when needed, unable to reserve the scanner, etc.
    e. * REJECT* - used to highlight images that are copied to one of the reject folders (RejectFolder, RejectOnLookupsFolder, RejectImageFolder)
    f. *TIME* - if 'hfServiceMgr=TRUE' then this will highlight when the time was exceeded during processing and barfile32.exe shut itself down in order to restart.
    g. *FATAL* - highlights an entry that is considered fatal to the successful running of Barfile32.exe, such as, DB1 value not being set up correctly inside barfile32.ini, or a problem trying to reset the scanner which would prevent successful barcode processing.
    h. *EXIT* - marks the point at which Barfile32.exe set 'hfServiceMgr_Restart=' because time was exceeded.
    i. *END* - indicates the shutdown of Barfile32.exe within the logfile.