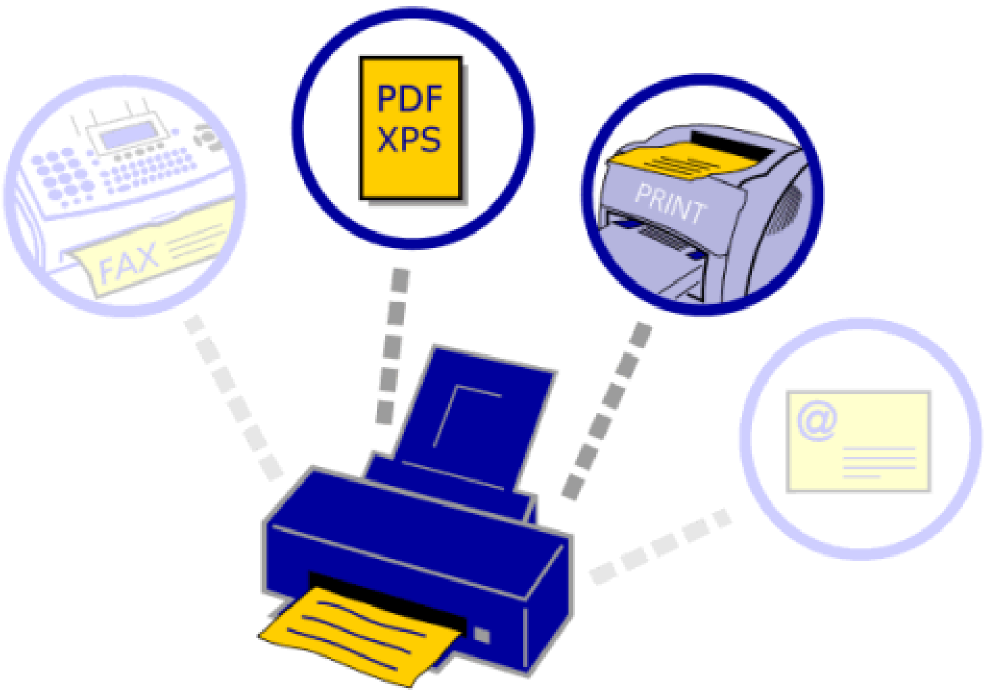


PrintMulti

(Print once – Produce multiple outputs)

Version 1.0.4.0

Dieter Riekert

http://www.lvbprint.de

info@lvbprint.de

Contents

Introduction

PrintMulti is a print processor, which makes it possible to output a print job to multiple printers. For each created job various settings can be applied.

Additional programs can be executed on the produced spool files.

Extensive, configurable log files are generated. With this option it is for example possible to detect which user has printed, for example, to a specific printer in color. The format for the log files is csv.

The print output can be modified in the following ways:

(Not all options can be combined)

Printing in reverse order

Several pages composed on one sheet (2, 4, 6, 9, 16), optionally with a border around the pages

Booklet printing with duplex printers

Simplex, duplex horizontal, duplex vertical

Changes to the paper format

Forcing Black-white / color printing

Different input bins for the first, subsequent and the last page

Saving the output into a file or append it to a file

Execution of programs on the generated files (e.g. for automatic pdf-creation)

All settings can contain conditions. This allows the user, for example to define printer output dependent on the number of pages, the paper format, the printing user, the document name, …

With the additional tool “devmode2file”, printer configurations can be saved to files and used with PrintMulti. This makes it possible to use output bins or watermarks, if the printer driver supports this.

New! You can use the printed document text to influence the configuration or use it inside scripts.

PrintMulti is controlled solely with configuration files in ini-fileformat.

***PrintMulti* was not designed for the unexercised user. It is intended for experts with knowledge in printing.**

License Agreement

This is a reduced translation of the original text in German language (from www.lvbprint.de)

Only the original text is legally binding.

Conditions of use:

You may install and use this software ("PrintMulti") without charge on any number of computers running under a Windows Client operating systems (2000, XP, Vista). This includes print servers.

To use the software on server operating systems (2000/2003 Server, Terminal Server) a license must be purchased. Installation for test purpose is granted.

The logging functionality is free for all kinds of operating systems, including servers.

To purchase server licenses please contact us.

Transfer the software:

You may distribute this software with your own software if you regard the conditions of use.

Limited Warranty and Limitation of Remedies:

The program and documentation are provided "as is" and without warranty, express and implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose. In no event will the author be liable for any damages, including lost profits, lost savings, or other incidental or consequential damages, even if the author is advised of the possibility of such damages or for any claim by you or any third party.

Conclusion:

The location of the competent court for all legal action in connection with the Software and this contract is D- Karlsruhe if the contract partner is a registered trader or equivalent, or if he has no legal domicile in Germany.

This contract is exclusively governed by the law of the Federal Republic of Germany.

Should any provision of the contract prove unenforceable or if the contract is incomplete, the remaining provisions will remain unaffected. The invalid provision shall be deemed replaced by the provision which in a legally binding matter comes nearest in its meaning and purpose to the unenforceable provision. This shall apply to any omission in the contract that may occur.

Changes

Version 1.0.4.0

Problems with the option "*Append2File*" solved.
Second independent log file was not created.

Version 1.0.3.3

New tools "*ShowPrinterInformation*"
New action type "*SaveEmf*"
Updated tool "*Devmode2File*" (long printer names)

Version 1.0.3.2

Use embedded fonts in spool files. Needed for PDF printing in some cases.

Version 1.0.3.0

Support for Windows 7 and Server 2008 R2 (32 and 64 Bit)
New macro "**p**" returns the standard printer of the printing user. Useful for Terminal Servers.
New operator „**rawpage**" returns a text representation of the printed pages. The text is not formatted. All strings are appended without any delimiters.
New operator „**tblsearch**" searches inside text for keys from configuration files. Can be combined with "rawpage" to allow document dependent printing.
New operator „**tbllookup**" looks up words in configuration files. Simplifies configuration with many fixed values and replaces nested conditional operators
New value „**WriteTextToFile**" for the „Print"-actions enables saving of the page content to a file. This file can be used inside scripts. A sample is contained.
New value „**RefreshConfig**" in „Exec"- and „Print"-actions forces to a reread of the configuration file after the current section. Enables dynamic changes to the configuration file.
Script „gscreatepdf.vbs" replaced by „**gscreatepdf1.vbs**" because of problems with spaces in file names.

Version 1.0.2.1

Recognizes the options "**Collate, Color, nUP, DrvCopies, TotalCopies**" for the PrintMulti assigned printer. Will also be used even if the section is inactive.

.

Technical requirement and limitations

Technical requirements

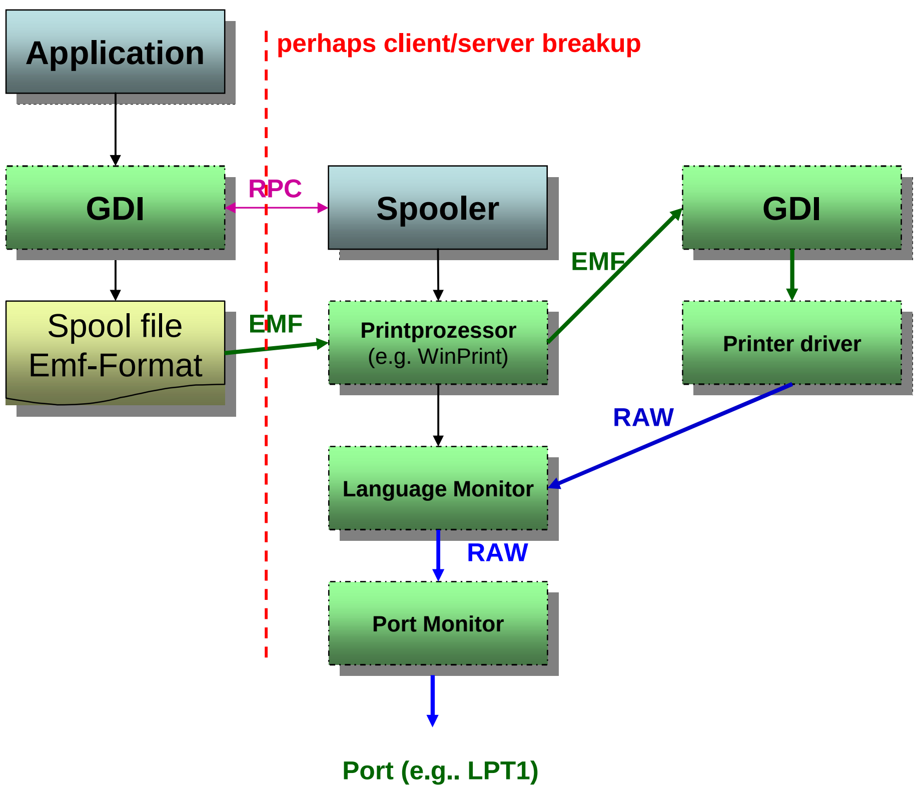
A print processor is a DLL, which the windows spooler calls to play back a spool file, recorded by the Windows GDI. The following link shows a good overview with all the components involved in Windows printing.

http://msdn2.microsoft.com/en-us/library/ms801074.aspx

The chapters below explain normal "EMF printing", the "RAW" printing mode and the mechanism that is used by PrintMulti.

EMF printing

EMF based printing

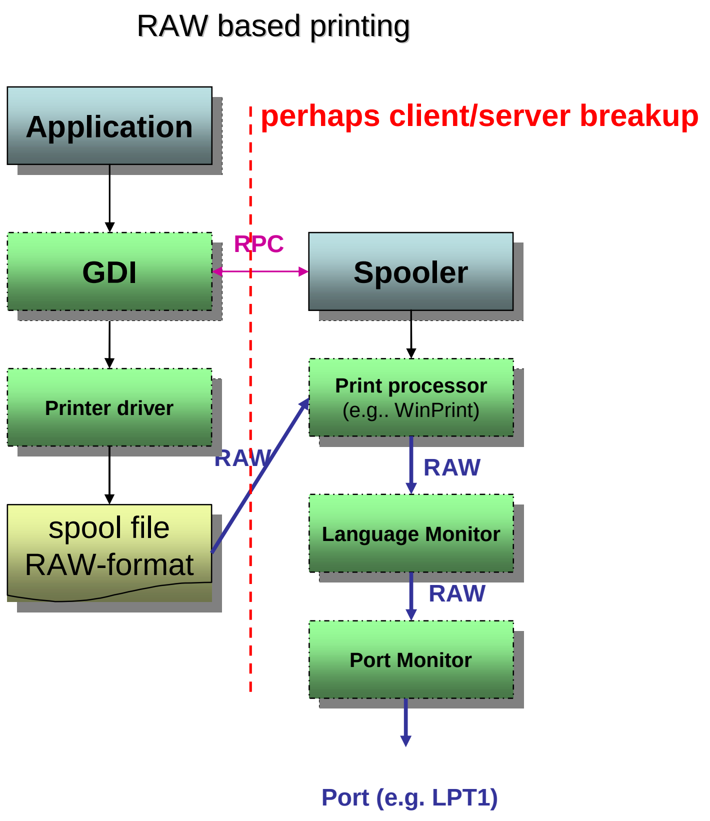


A printing application calls the GDI graphic commands, which are then written to a spool file (nearly unchanged as EMF records). The spooler will start with the spool job after it is activated by a RPC call. If the output is redirected to a network printer the spool file is then transferred to print server and the spooler on the print server then performs the task.

By using a spooler, the time intensive rendering of a "RAW" file is decoupled from the application.

This print-"path"is necessary for a successful use of PrintMulti.

RAW printing



Unfortunately there are situations where the render process is done in the first step by the GDI and the print processor receives the RAW file, which is stored identically in the spool file. Since the print processor receives all data in the destination format of the printer, PrintMulti can't perform the processing for which it is designed.

PrintMulti does not perform any RAW-processing. Please use other tools if needed (e.g. Redmon).

In the following case no EMF spooling in used:
(see also**: http://www.undocprint.org/winspool/spool_files#when_is_RAW_used**)

When print jobs are sent directly to the printer (printer configuration / extended)

The extended printing functions are not activated

Some drivers always print RAW (especially those from the latest printers direct from the homepages of the manufacturers). The drivers contained in Windows work in most cases.

If network printers are involved, the OS, EMF and driver versions must match (in the latter case deleting the client driver and reconnecting to the server will usually install the correct server driver versions)

The Adobe reader prints RAW to printers, which then use postscript drivers (performs a pdf to postscript conversion). Don't use a postscript driver for PrintMulti, if you plan to print from the Adobe reader.

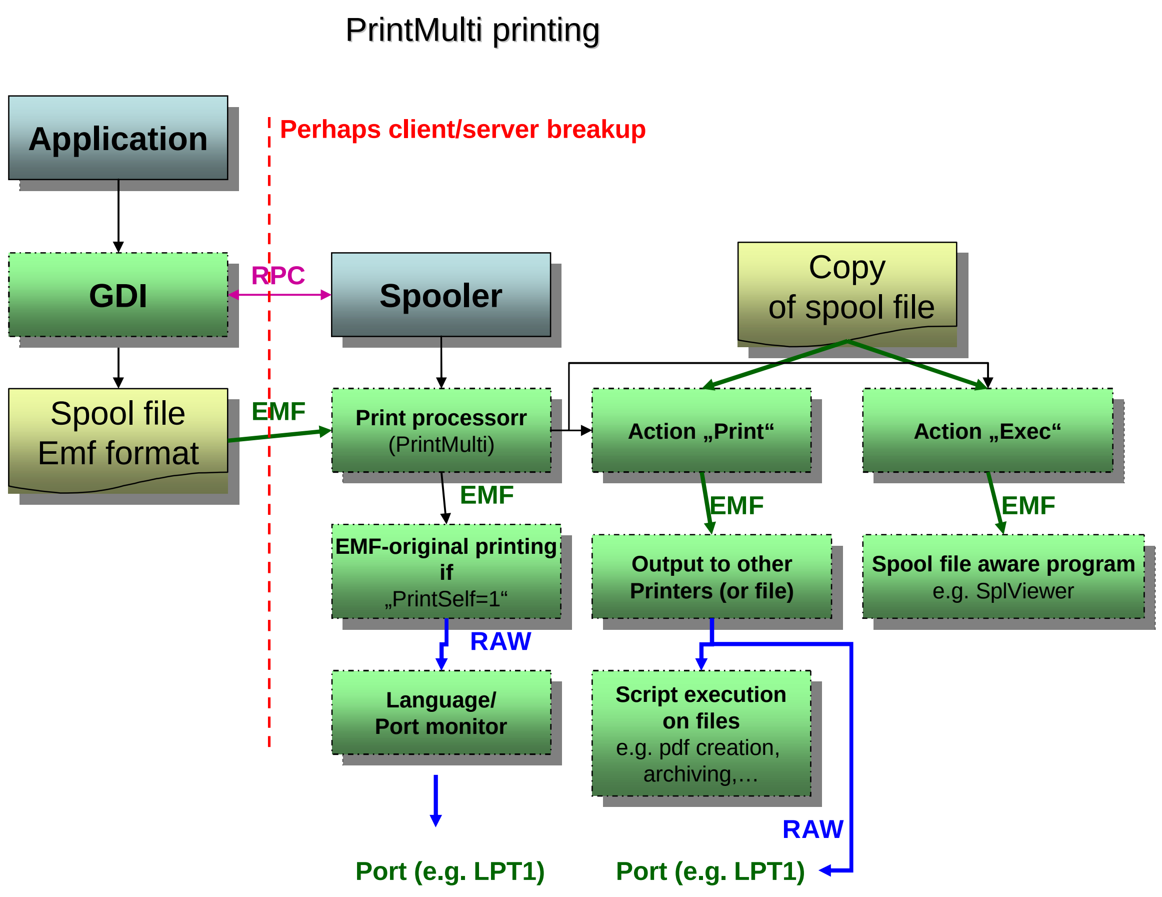
The referenced link above contains some additional cases.

If PrintMulti receives a RAW print job, it is forwarded to the port without any changes. The following entry is written to the log file in this case:

```
DEBUG   ;…;;;ATTENTION! PrintMulti does not support RAW printing. Job will go to the original printer without PrintMulti features.
```

You can prevent the RAW mode in most cases by using a dummy printer (see chapter ”the installation“)

PrintMulti printing


PrintMulti printing
Application
Perhaps client/server breakup
GDI
RPC
Spooler
Copy of spool file
Spool file Emf format
EMF
Print processorr (PrintMulti)
Action „Print“
Action „Exec“
EMF
EMF
EMF
EMF-original printing if „PrintSelf=1“
Output to other Printers (or file)
Spool file aware program e.g. SplViewer
RAW
Language/ Port monitor
Script execution on files e.g. pdf creation, archiving,…
RAW
Port (e.g. LPT1)
Port (e.g. LPT1)

PrintMulti copies the original spool file and performs specific actions on it. The original job can, in many cases be additionally executed (”PrintSelf=1“) if you use the drivers contained in Windows. The default for ”PrintSelf” is ”0“.

There are actions, which ”transfers“ the spool file through other printers and offer many options and scripting possibilities (”Print“-actions)

Other types of actions utilises programs, which understand directly the spool file. The SplViewer is an example for this kind of programs (”Exec“-actions)

Limitations and known problems

The following problems are known:

In many cases Windows creates a RAW print job. See chapter ”RAW printing“.
Solution: Use another printer driver *(e.g. HP Color Laserjet 8550 PCL) as PrintMulti printer*

The adobe reader creates sometimes temporary fonts during printing, which could be already deleted inside PrintMulti printing and lead to strange output. The problem seams to be dependent on the kind of font embedding in the pdf file (the best options for ghostscript are described later)
Solution: Convert *PDF or print PDF as picture (slow!) or use another PDF-Viewer (e.g. FoxIt).*

If the printer driver used for the PrintMulti printer leads to problems, use a driver contained in Windows. Select the ones with ”WinPrint“ as print processor.
Solution: Change printer driver *(e.g. HP Color Laserjet 8550 PCL)*

PrintMulti is based on copying and executing EMF files, which were created for one printer and should be printed to another. This is only possible, if the EMF files are device independent.
Unfortunately this is not always the case. One common case, that Excel inserts RAW PCL commands for thin line printing on PCL printers, is solved in PrintMulti. But there are certainly other unknown cases (there are many special postscript options which may influence the output).
Solution: Use another printer driver *(e.g. HP Color Laserjet 8550 PCL). Please contact us in this case.*

The PDF creation with the "ActionCreatePDFScript" may not work with printing clients (local printing is working).
The reason seams to be, that files created with the Option "Save2File" are not written for printing clients, although enough rights are present.
Solution: Try the option *"UseSystemAccount=1".*

Installation and configuration

The installation of PrintMulti is simple, usually done in a few seconds since only few small files are copied. The print processor (the “printmulti.dll“) is copied to the folder “%systemroot%\system32\spool\prtprocs\w32x86“ and the registry updated. For 64 Bit operating systems the destination folder is “…\prtprocs\x64”.

Additional files (manuals, scripts, the configuration file ‚printmulti.ini“) are always copied to folder ‚%programfiles%\printmulti“. A link in the Windows Start Menu is created for the manual.

Additionally an ICC profile with license information (from www.eci.org) is contained to create PDF/A files. Other free profiles can be downloaded from the above site. Color profiles are searched for in the folder ”%systemroot%\system32\spool\drivers\color“ by Windows. The script “gscreatepdf1.vbs” contains a reference to it.

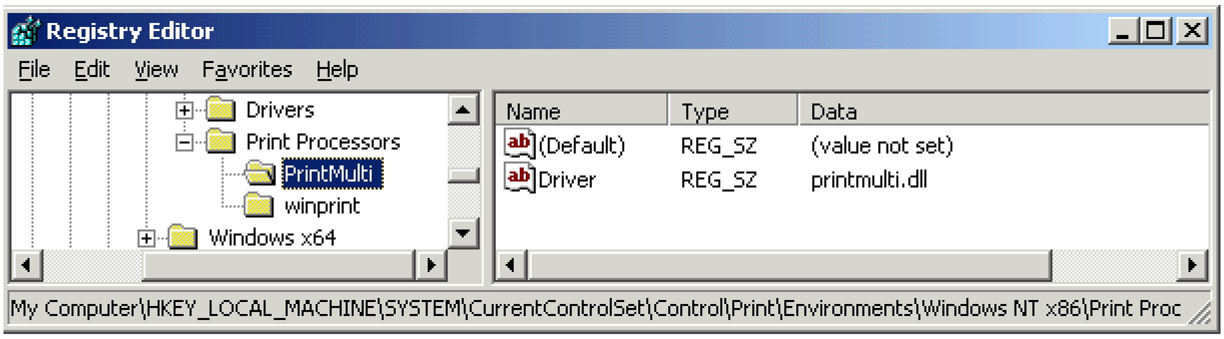
Existing files with the exception of the printmulti.dll will not be overridden. In case of an update, the new, possibly changed, sample configuration file is stored as “*printmulti.sample.in*”“.

For an update of PrintMulti the file ‚printmulti.dll“ must be replaced. Normally a restart of the computer will be necessary, since this file is in use. In most cases this can be prevented if the spooler is stopped with the command “*net stop spoole*”“ before the copy. It must be restarted with the command “*net start spooler*” after the installation. Take care that no print jobs are active. The installation program indicates when the spooler should be stopped.

The de-installation processdeletes all files with the exception of the ICC profile.
Be aware that the complete installation folder is removed, even if there are additional files there.

If you want to integrate PrintMulti in your software, copy the DLL to the folder “%systemroot%\system32\spool\prtprocs\w32x86”and call the Api-Function “AddPrintProcessor()”.

This will create the following registry entries:



Name	Type	Data
(Default)	REG_SZ	(value not set)
Driver	REG_SZ	printmulti.dll

My Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Environments\Windows NT x86\Print Proc

Add or remove the entries manually if required. The spooler service must be restarted in this case or the computer rebooted.

The installation file contains the versions for 32 and for 64 bit operating systems. The correct one is determined by a short startup program, which calls the native setup program.

The table shows an overview of the installed files and there description:

File name	**Path**	**Description**
Printmulti.dll	[1])	The print processor DLL
eciRGB_v2.icc	[2])	Simple color profile fort he PDF/A creation from www.eci.org
eciRGB_licence.rtf	[2])	License for the file above
printmulti.ini	[3])	Example configuration which is used after the installation.
Manual.pdf	[3])	This manual
Uninst.exe	[3])	The uninstaller that is called (also by System/AddPrograms)
devmode2file.exe	[3])	Allows saving device modes to files.
ShowPrinterInformation.exe	[3])	Shows (and saves) information about printers.
gscreatepdf1.vbs	[4])	Script for the PDF Creation with PrintMulti (replaces “gscreatepdf.vbs”)
PDFA_def.ps	[4])	Postscript file for the creation of PDF/A documents (contains replacement texts)
CmdRedirect.exe	[4])	Use command line parameters instead of „>“ for the redirection of the standard output channel. Used inside the “gscreatepdf1.vbs“ script.
TextExtract.vbs	[4])	Shows how to extract information from printed documents.
textextractsample.doc	[4])	Contains hidden information to be used with TextExtract.vbs.
SendMailExample.doc	[4])	Example to send an E-Mail (contains destination address)
Mailbody.txt	[4])	Contains mail body for the example
Blat.exe	[4])	Free commandline E-Mail application

Referenced folders:

[1]) %systemroot%\system32\spool\prtprocs\w32x86

[2]) %systemroot%\system32\spool\drivers\color

[3]) Installation folder of PrintMulti (%programfiles%\printmulti)

[4]) The folder (%programfiles%\printmulti\PDFCreation)

You should install or copy additionally:

> If you want to create PDF documents with Ghostscript, it has to be installed. Please start from version 8.61 (version before do not create 100 percent PDF/A conform documents). The script automatically detects the correct installation path
> http://downloads.sourceforge.net/ghostscript/gs861w32.exe
>
> For a full text extraction from the PDF file it is possible to use ‚pdftotext“. Please look at the end of ‚gscreatepdf1.vbs“ for an example how to call it.
> http://www.foolabs.com/xpdf/download.html

You should perhaps know, that the installation path of PrintMulti is stored in the registry at the location

”*HKEY_LOCAL_MACHINE\SOFTWARE\LVBPrint\PrintMulti\InstallPath*“

The environment variable “*PM_INSTALLPATH*” is set to this folder and available in scripts.

Printing with PrintMulti

All relevant data is loaded from a configuration file. The path and name of this file are stored in the registry at the key “*HKEY_LOCAL_MACHINE\SOFTWARE\LVBPrint\PrintMulti*”, Entry “*ConfigurationFile*”. If no entry is found, a file ”printmulti.ini“ is searched in the Windows directory.

The installation copies an example file to the path “<program files>\Printmulti” and sets the entry in the registry.

To test PrintMulti, we recommend installing a local printer, which prints to a single file (see the example)

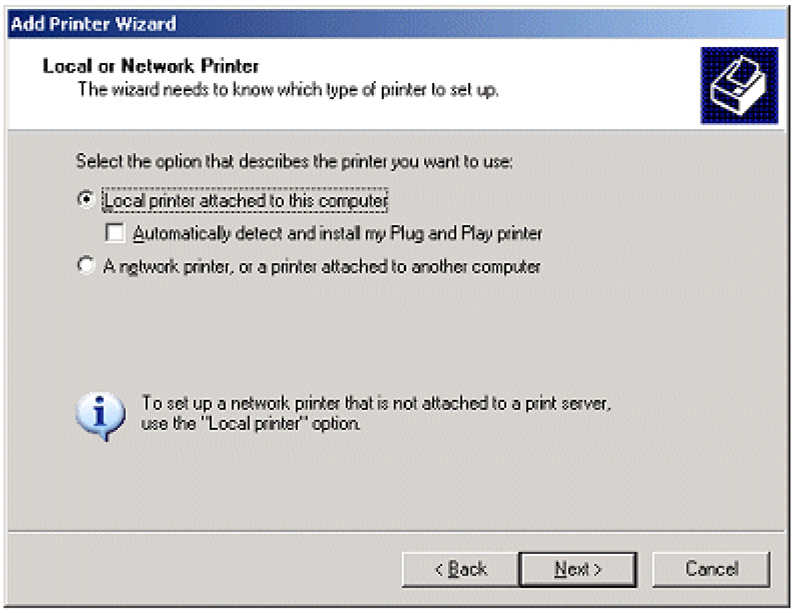
The following conditions must also be met to use PrintMulti:

1. The PrintProcessor ”PrintMulti“ must be assigned to the printer in the advanced options dialog.
2. The relevant configuration file must contain a section for the printer, the Entry ”*Active*“ must be set to ”1“ and at least one action has to be defined.

If nothing occurs when printing, check these conditions first

The following pictures show how to install a printer ”*MiniPhotos*“, which causes a print job to be sent to the printer ”*PhotoPrinter*“ twice. The first job is printed with unchanged settings and the second one with nine pages compressed on one sheet in black/white.

Some intermediate steps are suppressed.


Add Printer Wizard

Local or Network Printer
The wizard needs to know which type of printer to set up.

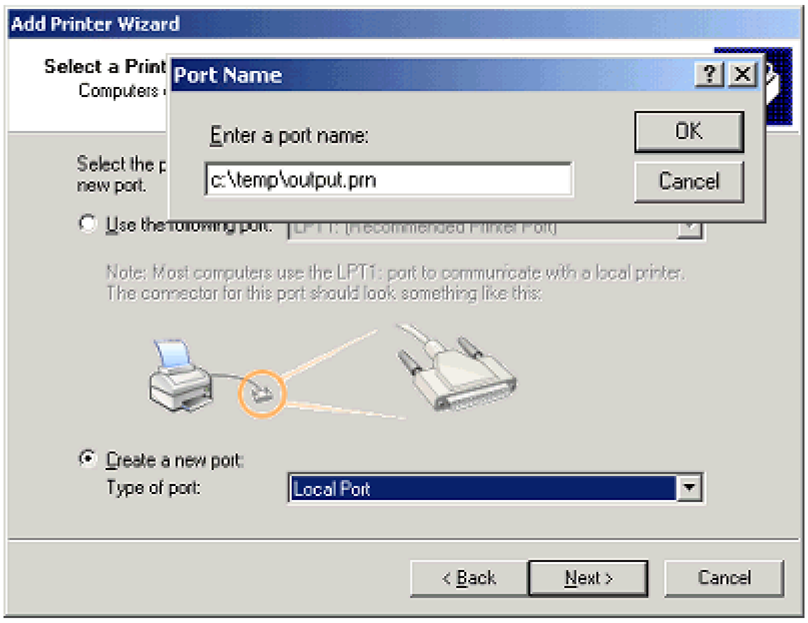
Select the option that describes the printer you want to use:

(●) Local printer attached to this computer
☐ Automatically detect and install my Plug and Play printer
( ) A network printer, or a printer attached to another computer

To set up a network printer that is not attached to a print server, use the "Local printer" option.

< Back | Next > | Cancel

Install a new printer


Add Printer Wizard

Select a Print
Computers

Select the p
new port.

( ) Use the following port: LPT1: (Recommended Printer Port)

Note: Most computers use the LPT1: port to communicate with a local printer. The connector for this port should look something like this:

(●) Create a new port:
Type of port: Local Port

< Back | Next > | Cancel

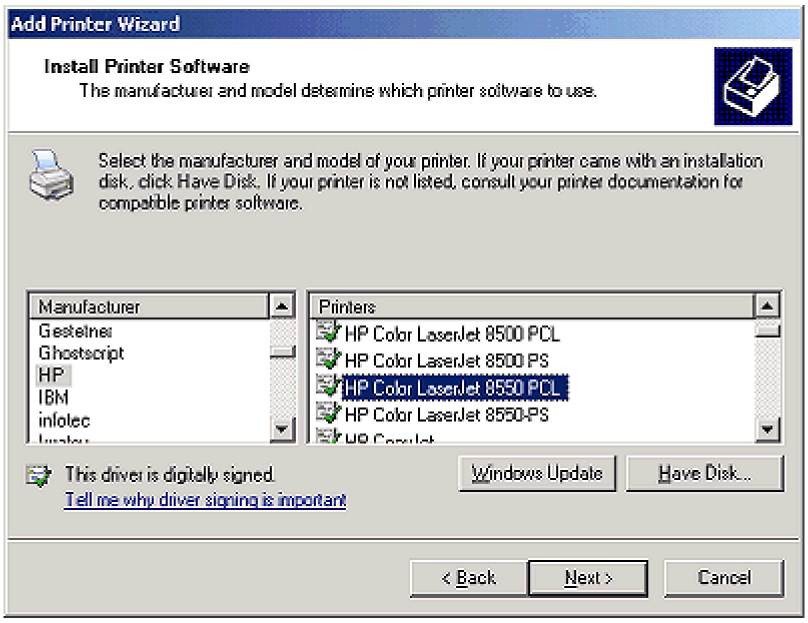
Port Name

Enter a port name: c:\temp\output.prn

OK | Cancel

As port you can use a new ”Local Port“ with a fix file name as destination (If you share the printer all users must have write access to this file!). All output to the printer is saved to this file overwriting the content each time.

After the printer is configured with ”PrintMulti“, there will be no print output to this printer anymore.



Add Printer Wizard

Install Printer Software
The manufacturer and model determine which printer software to use.

Select the manufacturer and model of your printer. If your printer came with an installation disk, click Have Disk. If your printer is not listed, consult your printer documentation for compatible printer software.

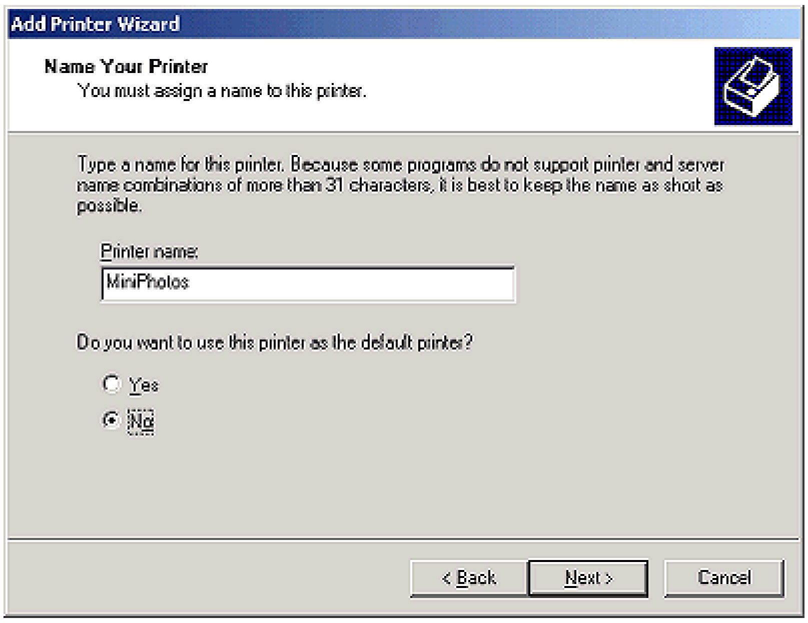
Manufacturer
Gestetner
Ghostscript
HP
IBM
infotec

Printers
HP Color LaserJet 8500 PCL
HP Color LaserJet 8500 PS
HP Color LaserJet 8550 PCL
HP Color LaserJet 8550-PS

This driver is digitally signed.
Tell me why driver signing is important

Choose a printer which supports many paper format, color printing and duplex.

Do not use Postscript drivers if you plan to print PDFs with Adobe Acrobat Reader
(➔ Raw Printing)



Add Printer Wizard

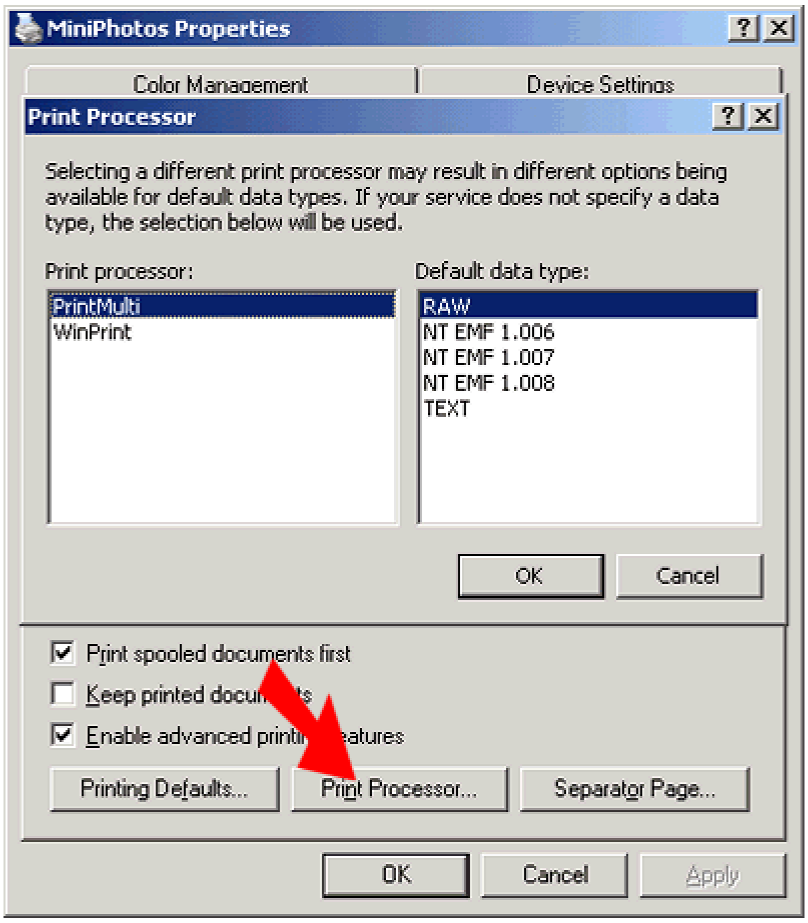
Name Your Printer
You must assign a name to this printer.

Type a name for this printer. Because some programs do not support printer and server name combinations of more than 31 characters, it is best to keep the name as short as possible.

Printer name: MiniPhotos

Do you want to use this printer as the default printer?

Yes [✓ No]



Selecting a different print processor may result in different options being available for default data types. If your service does not specify a data type, the selection below will be used.

Print processor: PrintMulti (selected), WinPrint

Default data type: RAW (selected), NT EMF 1.006, NT EMF 1.007, NT EMF 1.008, TEXT

[✓] Print spooled documents first
[ ] Keep printed documents
[✓] Enable advanced printing features

Do not forget to enable the duplexing feature if you want to use duplex mode.

”*PrintMulti*“ can now be set as print processor. The data type can stay ”Raw“ as suggested.

```
[Common]
LogJobMask=4
LogJobFile=d:\temp\jobs.csv

LogMask=27
LogFile=d:\temp\debug.csv

[MiniPhotos]
Active=1
Action1=Print;ActionPhotoPrinter1Co
Action2=Print;ActionPhotoPrinter9BW

[ActionPhotoPrinter1Co]
Active=1
Printer=PhotoPrinter
nUp=1

[ActionPhotoPrinter9BW]
Active=1
Printer=PhotoPrinter
Color=0
nUp=9
nUpBorder=1
```

A minimal configuration file

”C:\Program Files\PrintMulti\printmulti.ini“ is shown

Two log files are written – one with the job data and one with extended debugging information.

Each job sent to the printer ”MiniPhotos“ then has the two actions performed in the given order, if ”Active“ is set to 1 (there will be no output to the printer MiniPhotos even, if it is attached to a physical port. Exclusion: ”PrintSelf“ is defined).

If ”Active=0“ (in MiniPhotos section) the printer will behave as if no PrintMulti is installed. But Logfiles will be created anyway.

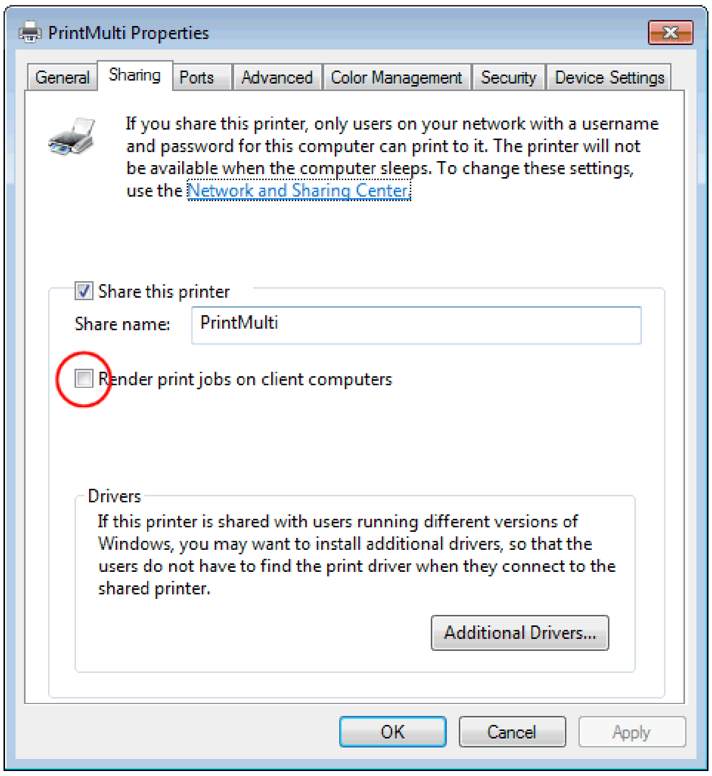
Sharing of PrintMulti-printers

As mentioned earlier, Print Multi requires spool files in EMF format to work correctly. Especially in a client/server environment Microsoft has changed the preferred printing architecture and propagates now the "Client-Side Rendering" (CSR) (http://blogs.technet.com/b/askperf/archive/2008/02/10/ws2008-client-side-rendering.aspx)

With CSR enabled, the client renders the job and sends the result to the spooler on the server. The data format is the RAW format of the spooler (e.g.: PCL)

CSR is configured in the sharing dialog of the printer. Accordingly to the article above, there are cases where the setting is ignored in both ways.

Sharing a printer with Vista, Windows7 or Server 2008 requires deactivating the CSR setting



If you share this printer, only users on your network with a username and password for this computer can print to it. The printer will not be available when the computer sleeps. To change these settings, use the Network and Sharing Center.

[✓] Share this printer

Share name: PrintMulti

[ ] Render print jobs on client computers

Drivers

If this printer is shared with users running different versions of Windows, you may want to install additional drivers, so that the users do not have to find the print driver when they connect to the shared printer.

If you connect printers from Windows 7 clients on older servers (2003) or clients (XP), than you have to set the CSR option at the properties dialog of the connected printer. The server is not offering the option, but the client seams to use it.

Attention!
There seams to be problems with the writing of files using the option "Save2File" on the server side. This way is used when creating PDF files with Ghostscript ("ActionCreatePDFScript").
If you notice such problems, try the option "UseSystemAccount=1".

You can also use 64 bit clients with 32 bit servers and the other way around. That works fine, if you use a driver based on the UNIDRV drivers from microsoft and you add the drivers in the sharing dialog (see above "Additional Drivers".

You need some files for the corresponding architecture (stdnames.gpd, ttfsub.gpd, unidrv.dll, unidrv.hlp, unidrvui.dll, unires.dll). The dll types must fit (if you add a 64 bit driver then the DLLs must also be compiled for 64 bit).

Macro expansion and calculable expressions

All settings in the configuration file can contain macros and conditional expressions based on a simple stack calculation machine (like old HP calculators).

These new features make PrintMulti very flexible. The examples (shown later) show how to achieve the following:

Paper format dependent printing (A3 jobs to an A3 printer, A4 jobs to an other)

User dependent printing (e.g. only some persons may print in color)

Printing dependent on the number of pages (e.g. print documents with more than 1000 pages to a faster printer)

Document name dependent printing (e.g. print documents with a special name to different printers)

Time dependent printing (print to a silent printer at night)

The ungenerous user could use a single printer for all this purpose.

Macro expansion

Macros are shortcuts for known data values of the print job, the configuration or environment variable. The previously mentioned ”device mode“ contains settings, which the user selects when printing a document from an application, like the paper format, the duplex mode etc …

The device mode includes many standard settings that can be accessed by macros and other driver specific values that cannot be accessed (like the output trays).

An example from the configuration of the log file, which contains many examples:

”#T;#(06)J;#(-20)P;#(07)C;#(-20)U;#(-20)M;#(-30)D;#(5)Z;#(5)c;#(-6)B;#(-10)A;#(-20)E“

This string will be used for each debug output, if not defined otherwise.

The following rules apply for string substitution:

A macro starts with ‚#“ optional followed by a format expression in brackets and terminated by a letter e.g. ”#(…)Z“ or

An environment variable surrounded by the “%“ character e.g. *”%TEMP%“*.
Only system environment variable can be used (no user environment variable).

The table shows the different macro letters.

Macro	Type	Meaning
Data from the print job and PrintMulti configuration		
A	String	Name of the action (e.g. ”ActionPhotoPrinter1Co“ in the example above).
B	String	The type of the action (”print“ or ”exec“ at the moment).
C	DWORD	Printer specific counter (for the Printmulti aware printer). This value will be increased with each print job. It is stored in the registry and must be manually reset.
D	String	The name of the document, which is set by the printing application.
K	String	The name of the document with invalid characters for path names replaced by an underline. Invalid characters are: (<>:“/|). Use this instead of ”D“ if the macro is part of a file name.
E	String	Action specific text, which will be replaced with the name of the destination printer in a ”print“-action. (e.g.: ”PhotoPrinter“).
F	String	File name of the internally copied spool file.
G	String	Action specific file name. In a ”print“-action, when output goes to a file, this file name will be used (”Save2File“). Otherwise the name of the spool file is used (=“F“).
g	String	Like ”G“, without file extension (e.g. ”c:\temp\file“).
I	String	Use the expression from another configuration section („Include“)
J	DWORD	Job id of the print job (not unique, will be reused by windows).
M	String	Name of the computer which starts the print job.
P	String	Name of the main (PrintMulti) printer (e.g. ”MiniPhotos“ from above).

p	String	**New:** Standard printer of the printing user (will not work in all cases - especially for shared printers)
S	Date/Time	Time, when the job was created.
T	Date/Time	Current time.
U	String	Name of the user who printed.
Z	DWORD	Number of pages (from the job)
z	DWORD	Total number of copies (only for action ”Print“).
Data from the device mode of the print job		
c	0/1	color, 0 = black/white, 1 = color (dmColor – 1)
b	DWORD	Paper source (dmDefaultSource)
d	0/1/2/3	0 = not set, 1 = simplex, 2 = duplex vertical, 3 = duplex horizontal (dmDuplex)
o	0/1/2	0 = not set, 1 = portrait, 2 = landscape (dmOrientation)
s	DWORD	Paper size (e.g.. 8 = A3, 9 = A4) (dmPaperSize), 0 = not set
y	DWORD	resolution (dmYResolution), 0 = not set

The permitted format expressions are dependent on the “type” in the table above. The following formats are configurable for the different types. To understand them, you must know, how the printf format works inside the C-programming language.

String:

Example:	”myTest“
Standard:	existing string, e.g. ”#D“ ”myTest“
Format:	Embedded in printf with ”%<format>s“, e.g. ”#(-10)D“ is replaced by ”%-10s“, in the example ”myTest ”. The characters ”%“ and ”*“ are forbidden in the format string and lead to a standard format.

DWORD or number:

Example:	40000
Standard:	”%u“ e.g. ”#C“ ”40000“
Format:	If the last character in the format string is one of ”diuxXo“, then it will be used for formatting. Otherwise, ”%u“ is used. ”#(08X)C“ ”%08X“ the example will expand to ”00009C40“, “#(08)C“ ”%08u“ expands to ”00040000“. ”%“ in the string is forbidden and lead to a standard format.

Date / Time:

Example:	1964-11-23 13:45:12
Standard:	from ”printmulti.ini“ or ”%Y-%m-%d %H:%M:%S“, if not configured
Format:	will be formatted with the function ”strftime“. Please search the internet for a description.

Calculable expressions

Common rules

Each configuration entry can contain calculation expressions on the right side, which are evaluated during each access of the key. Other entries can be referenced with the new macro ”#(<Section.Key>)I“ which increases the clearness.

The following rules are used:

A calculation expression must be in the format: ”$(<expression>)“, where the expression is defined as ”<Operand/Operator>;<Operand/Operator>;…;<Operand/Operator>”.

An expression is composed of operands and operators, which are separated by a semicolon. Operands are pushed to the stack. Operators pop the according number of values from the stack, calculate the result and push that back to the stack.

At the end of a calculation exactly one value must be contained in the stack, otherwise an error is generated.

Operands must be enclosed by quotes if they contain operators or special characters like a semicolon or the closing bracket.

Blanks are removed at the start and end of an operand, if they are not enclosed in quotes.

An operand is considered as a number, if it solely consists of digits (negative numbers are not supported at the moment). Many operators have a different meaning for numbers and strings. An arithmetic operation is only performed, if all operands are numbers (e.g.: ”$(2;3;+) evaluates to ”5“, $(2;A;+)“ to ”2A“)

Use the flag with the value of ”32“ in the common section of the printmulti.ini to output debug messages about calculation expressions.

An expression can contain macros which are valid for the configuration entry.
(e.g..: ”LastPage=$(#Z;1;-)“, sets the last printed page to number of pages subtracted by one; means: do not print the last page)

Each expression is recalculated on use.

For operators which are expressed as text (and, or, not, contains, upper, lower, tblsearch, tbllookup) are not case sensitive (”AnD“ will be accepted)

If you use the page text (with „RAWPAGE“) try the option “WriteTextToFile”, if something is not working. It does not work for all printing applications and the parts to search must be inside the visible page that is printed and must not be cut.

Operator table

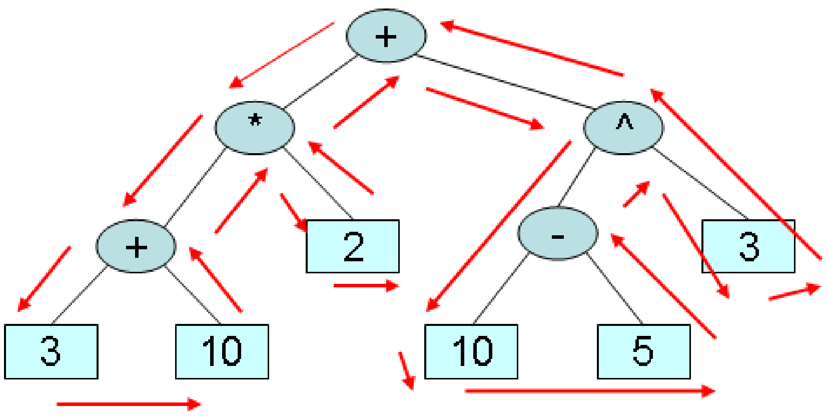
Op	Description	¹)	Example
+	Arithmetic addition, if both operands are numbers otherwise string concatenation	S	$(2;3;+) 5 $(abc;4;+) abc4
-	Subtraction (numbers only)		$(100;5;-) 95
*	Multiplication (numbers only)		$(100;5;*) 500
/	Division (numbers only) – no rounding		$(101;3;/) 33
<,	An arithmetic comparison is performed, if both operands are numbers. In the other case a string comparison is done independently from the current locale setting (see ”wcscmp“). For the two last operators there are two equal shortcuts. Blanks outside quotes are eliminated for the comparison.	S	$(A;B;<) 1
>,		S	$(A;a;>) 0
<=,		S	$(“ Beta“; Alpha;<) 1 Blank before “B“ < ”A“
>=,		S	
==,=		S	$(A;A;==) 1
<>,!=		S	$(“ A“; A;!=) 1
!, not	Returns 1, if the value = 0		$(A;B;<;not) 1 $(4711;!) 0
and	Logical operators need two numeric operands. A number ”!= 0“ is considered as logical 1.		$(1;2;And) 1
or			$(0;0;or) 0 $(9;0;or) 1
upper	Conversion to uppercase letters using the current locale setting.	L	$(“äöüß“;upper) <u>Log file:</u> Upper(“äöüß“)->”ÄÖÜß“ Locale:German
lower		L	
contains	”1“, if the second text is contained in the first one. ”Case sensitive“ comparison.	S	$(ABCDEF;cd;contains) 0 $(abCdE;upper;cd;upper; contains) 1
?	The C-Conditional operator; <Condition> ? <True-Value> : <False-Value> $(<Condition>;<True-Value>;<False-Value>;?)		$(A;B;<;1;2;?) 1
rawpage	Receives the page number as optional parameter and returns the page content as text (without any delimiters). Without the page number, all pages are returned, delimited by CR/LF.	S	`$(1;rawpage)    text of the first page` `$(;rawpage)    all pages` `More examples below`
tbllookup	Searches for a key in a section, which matches the given value. The search is case insensitive. If no key is found, the default value is returned. If “<File>“is empty, then the actual configuration file is used. Files without folder parts are searched in the System32-path. Use %PM_INSTALLPATH% if desired as path to search in the PrintMulti configuration folder. $(<File>;<Section>;<Key>;<Default>;tbllookup	S	`[Users]` `adam=$(#Z;100;<)` `gustav=1` `[ActionPrint]` … `Color=$(;Users;#U;0;tbllookup)` … Gustav prints in color; Adam only for documents with less than 100 pages
tblsearch	Other lookup strategy as for „tbllookup“. All keys in a section are taken one by one and used for a search in a string. If the key is found (“case sensitive search“ in opposite to tbllookup) the value on the right side is calculated and returned. The file name is examined as above. $(<File>;<Section>;<String>;<Default>;tblsearch	S	`[Printers]` `#invoice#=Printer1` `#delivery#=Printer2` `[ActionPrint]` … `Printer=$(;Printers;1;RAWPAGE;#p;tblsearch` … A Word document could contain on the first page („1;RAWPAGE“) for invoice printing anywhere the text „#invoice#“. If none of the both values are found, the user standard

			printer is used (#p).

[1])	S	string relevant functions
	L	Uses actual locale settings from windows

Example conversion of an expression into a stack based expression

(3 + 10) * 2 + (10 – 5) ^ 3



When traversing the parse tree, the left children are first noted, then the right ones and the parent node at the end. The red arrows mark the traversing order of the nodes. The example is represented by the following stack based expression:

3 10 + 2 * 10 5 – 3 ^ +

Examples to this subject area are found later at the common examples.

Sections and actions in the ”printmulti.ini“

The configuration file will be reread with every print job. Changes will immediately affect the next print job.

Common

The common-section is mostly used to set options for the logging output. It is possible to handle two independent log files. The file names can contain macros from the last section with the exception of device mode macros.

e.g. ”C:\LogFiles\#P\#U_#(%Y_%m).csv“ creates for each printer and each user every month a new file. In the example before: ”C:\LogFiles\MiniPhotos\dieter_2007_05.csv“.

Most of the possible values are strings, which can contain macros. All characters, which do not belong to a macro, will be used unmodified. Additionally it is possible to define the date/time-format, which will be used with the macros ”#T“ and ”#S“ if no format string is given.

For the three output masks definitions the bit field has the following meaning:
(Add the desired values and use the sum)

0	No output
1	Error
2	Warning
4	Job info
8	Info
16	More info
32	Calculations

Example:

```
[Common]
; Bitmasks:1=Error,2=Warning,4=Jobs;8=Debug;16=More Debug;32=Calculations
; Used with DbgView from sysinternals
DbgOutMask=255

; Two separate debug files, each with a different bitmask. No file -> No output
LogJobMask=4
LogJobFile=d:\temp\jobs.csv
LogJobFileFormat=#T;#(06)J;#(-20)P;#(07)C;#(-20)U;#(-20)M;#(-30)D;#(5)Z;#(5)c;#(-
6)B;#(- ►10)A;#(-20)E
LogJobHeader=Type    ;Date/Time          ;JobId ;Printer             ;Counter;User
►;Machine             ;Documenttitle
►;Pages;Color;Action;ActionName;Actionmessage       ;JobMessage
LogJobHeaderOut=y

LogMask=27
LogFile=d:\temp\#P\debug.csv
LogFileFormat=#T;#(06)J;#(-20)P;#(-5)B;#(-10)A
LogFileHeader=Type    ;Date/Time          ;JobId ;Printer
►;Action;ActionName;Message
LogFileHeaderOut=y
LogStdDateFormat=%Y-%m-%d %H:%M:%S
```

The lines beginning with ”►“ are the continuation of the last line and are only wrapped here.

(The LogJob… and LogFile… entries only have different names. They behave the same. The table lists only one type. The standard entries are shown in the example above.

The table shows the meaning of the available settings:

Key	Standard	Description
DbgOutMask	0	Bitmask for output messages to an attached debugger
LogJobMask	0	Bitmask for output messages to a file
LogJobFile	””	Output file. If no valid filename is defined nothing will be written!!
LogJobFileFormat	See example	Format for debug messages
LogJobHeader	See example	A header, which will be written as first line in new files, if the next value is ”true“
LogJobHeaderOut	True	Output the header or ignore the ”LogJobHeader“ entry
LogStdDateFormat	See example	Date/time format used, if no other format is given in time relevant macros

The debug message itself will be appended to the text defined with ”LogJobFileFormat“ or ”LogFileFormat“.

If the debugmask contains the bitmask value ”JobInfo“, then for each print job a line is added with the status ”FAILED“ or ”OK“, followed by the action name ”Print“, ”Exec“, … and the duration of the job in ms.

For the main print job a debug entry is added after all action entries. It contains ”OK“, if all actions were successfully. Action type and name are empty for this entry.

Entries for the PrintMulti-Printer

Entries for the PrintMulti-Printer			
Key	**Type**	**Default**	**Description**
Active	Bool	0	Use PrintMulti at all?
PrintSelf	Bool	0	Output to the physical printer (1) or just execute the contained actions (0)
Action*	String	-	Actions are executed in the order they appear.
Collate	Bool	-	Overrides collate setting for original job.
Color	Bool	-	Overrides color settings for the original job
nUp	Int	-	Overrides number of pages per sheet
nUpBorder	Bool	-	Use border for nUp
DrvCopies	Int	-	Overrides number of copies
TotalCopies	Int	-	Overrides total number of copies

As mentioned earlier, if you want to use the ”*PrintMulti*“-features, you have to use ”*PrintMulti*“ as print processor, create a section in the configuration file with the name of the printer and set the entry ”***Active***“ to true (=1 or can be a text which starts with a ”y“ or ”j“ character). The default for ”Active“ is false (=0 or a text beginning with ”n“).

In this case the original function (winprint) will be called (recognizing the new six settings!). The log entries will still be created. It is possible to use printmulti solely as log output medium, even for Windows 2003 Server without a license.

If ***”PrintSelf“*** is set to true, then output of the print job will be sent (in addition to the actions) to the configured port of the printer with all original settings, recognizing the six new settings. In the other case no output is sent to the original port (= default behavior). This features works best for printer drivers that are contained in Windows and use ”winprint“ as print processor.

The printer section may contain action entries, which are executed in the order they are listed in the file. An action entry has to start with the word ”***Action***“ (case insensitive). The rest is arbitrary. On the right side of the equal sign you define the type and the section of the action. The action will only be executed, if the entry ”***Active***“ is set to true in the action section. That is also the default for this entry (opposite to the default in the printer section).

”Collate“, ”Color“, ”nUp“, ”nUpBorder“, ”DrvCopies“, ”TotalCopies“ are new options for PrintMulti starting at version 1.0.2.1. They allow overriding of the settings, which the user has decided for the print job. The meaning is the same as for the print section described later. If no value is configured, the original setting is used. They are used, if ”PrintSelf“ is set or PrintMulti is deactivated with ”Active=0“.

```
[PrintMultiPrinter]
Active=1                                  (Default:“0“)
ActionAnyTextHere=Print;Section1          (will be executed)
Action2=Print;Section2                    (will be executed)
Action3=Print;Section3;1                  (will not be executed – Active is 0)
Action4=Print;Section1;0                  (will not be executed)
PrintSelf=1                               (output original job)
Color=0                                   (prints b/w, overrides user setting)
[Section1]
Active=1                                  (Default: ”1”)
…
[Section2]
…
[Section3]
Active=0
```

The ”Print“ action

The following adjustments affect the action orientated printing:

Action Print			
Key	**Type**	**Default**	**Description**
Active	Bool	1	Use the action?
Printer	String	Section name	Printer name to print to
UseSystemAccount	Bool	0	Print with the system account (normally not necessary).
nUp	Int	*0)	Number of pages per sheet. (1,2,4,6,9,16)
nUpBorder	Bool	*0)	Border around the pages
Booklet	Bool	*0)	Booklet printing
Duplex	String	Simplex	”s“ or ”S“ for Simplex, ”v“ or ”V“ for Duplex vertical and ”h“ or ”H“ for Duplex horizontal
Reverse	Bool	*0)	Print last page first. Does not work with all other options.
Collate	Bool	*0)	Change page order for multiple copies: 1-2-3-1-2-3 in place of 1-1-2-2-3-3
Color	Bool	From device mode	Allows to force black/white printing for color printers
DrvCopies	Int	*1)	Number of copies to set in the device mode
TotalCopies	Int	*1)	Total number of copies
FirstPage LastPage	Int Int	1 No. of pages	Page range to print
Devmode1	String	-	Device mode from file
PaperSize PaperSizeConversion PaperSizeConvertAlways	Int String Bool		Paper size relevant settings
PaperSource1 PaperSourceN PaperSourceL PaperSourceConversion	Int Int Int String		Paper source for different pages
Save2File	String	””	Print output will be saved to the file. The file name can contain macros and calculations
Append2File	Bool	False	Append to the defined file

Execute ExecuteCmd ExecuteCurDir ExecuteAddPath ExecuteFlags ExecuteTimeout ExecuteAsUser ExecuteShowWnd	Bool String String String Int Int Bool Int	False - - - 0 Not waiting False Normal	Options for the execution of programs on the created RAW-files
WriteTextToFile	**String**	**-**	**Writes the extracted text from the printed document to the given file**
RefreshConfig	**Bool**	**False**	**Rereads the configuration file after the current secion. Useful for dynamic changes to the configuration file**

*0) The standard for these values are determined from the print job attributes. Some drivers allow setting the number of pages and the activation of booklet printing (only for duplex printers) in the printer options.

*1) The number of copies should not always be taken from the main job. If you plan to use PrintMulti for concurrent archiving for example, there is no point in storing multiple copies in the archive.

For this purpose the two settings ”DrvCopies“ and ”TotalCopies“ are useful. If not defined both values are copied from the job data. ”TotalCopies“ specifies the complete number of copies. ”DrvCopies“ the value that is written to the device mode at each loop.

Example: TotalCopies 7, DrvCopies 3 lead to three loops with 3, 3 and 1 copy in the device mode.

PaperSize-Setting

In many cases the slave printers do not share the same printer formats with the main printer. If for example the user chooses DIN A3 format for the print multi printer and the slave printer is not able to print this format, then PrintMulti is unable to perform the correct function.

Configuration settings must then help PrintMulti to solve this problem.

A setting of “PaperSize” causes the use of that paper format in any case independent of the one set at the main printer. More flexible is the use of the setting ”PaperSizeConversion“. It is possible to define a mapping from source formats to destination formats.

The syntax is the source format followed by ”->“ and the destination format. Multiple entries can be separated by semicolon. A source of ”0“ defines the default format, which is used if no other format fits.

Format values can be the format names or the corresponding windows constants (see addendum).

If no identical text is found, the first entry which starts with the text is taken.

Example: ”A3->Exe;0->A4;1->Letter;A4->B5“
Default is ”A4“; ”Executive“ is used for the first setting

If ”PaperSizeConvertAlways“ is set to true, the mapping is applied in any case. In the there case only if the format is not available in the destination printer. Remember, that the default value is used, if formats could not be found in the mapping.

PaperSource-Setting

The selection of paper trays is very similar to the paper formats selection. If values are set for the first page, the following pages and/or the last page, these values are used in every case (for a single page ”PaperSource1“ is taken; for two pages ”Papersource1“ and ”PaperSourceL“).

Settings affect the following pages (e.g. if only ”*PaperSource1*“ is defined, than the value is used for all pages).

Unfortunately the numbers and paper names are dependent of the printer driver. This complicates the definition of the mappings. Names are always searched in the context of the destination printer. First a match for the exact name is searched and if not successful a search for identical text beginning performed.

Example: ”*manu->lower;3->2;15->auto*“ (replaces ”manual“ and ”automatic“ for many drivers)

Execute-Settings

Activating ”*Execute*“ causes a process execution after the printing. If the output is configured to go into a file (”*Save2File*“), the macros ”#G“ and ”#g“ contains this file name. Both macros are filled with the name of the spool file, if there is no file output.

The process to be executed must be configured with the ”*ExecuteCmd*“ setting. The actual path can optionally set with the variable ”*ExecuteCurDir*“.

”*ExecuteTimeout*“ controls the wait for the end of the called process. If the value is > 0, then the calling thread waits the amount in milliseconds and then continues even if the process is still running. If ”*INF*“ (=INFINITE) is set the calling thread will wait until the end of the process. A value of ”0“ causes the thread not to wait at all.

The possible values of ”ExecuteFlags“ and ”ExecuteShowWnd“ can be looked up in the description of the Windows-Api-function ”CreateProcess“. A common value for ”ExecuteFlags“ is ”0x08000000“. In this case no console window is displayed when calling a batch file (don’t forget to set the value back if you have problems and need the console output).

If ”ExecuteAsUser“ is configured as false, then the process is executed in the context of the user ”System“. In the other case the printing user is used, which is the normal case in Windows.
”ExecuteShowWnd“ will be 1 in the normal case (SW_SHOWNORMAL).

Some environment variables are available in the called process. The values are listed in the addendum. Additionally the data contained in ”*ExecuteAddPath*“ are appended to the ”PATH“. All configuration variables that starts with a ”!“ are additionally included in the environment (macros and calculations are performed).

Warning!!! There is no registry path ”HKEY_CURRENT_USER“ for the user available. Registry data is read from the folder ”HKEY_USERS\.DEFAULT“. But do not rely on it. If you need settings for the user (e.g. for database or network access) you have to copy the values to that location. You can use ”regmon“ from http://sysinternals.com to monitor the real access.

Text extraction

From version 1.0.3.0 it is possible to use the content of the printed document for configuration purpose.
Does not work for all applications (e.g. PDF printing)

At the moment only the unformated text is available. All text output from the application is stored in sequential order in a string without any delimiter. A page change leads to a carriage return/line feed.

There are two possiblities to access the text:

On the one hand the option "*WriteTextToFile*" allows to save the data to a file in a print action. This is done before executing scripts with "*ExecuteCmd*", allowing a script to read the file and use the content. The parts to used should be marked in a special way (e.g.: surrounded by special characters) and can also be very small and printed invisible with white color.
On the other hand the macro "*rawpage*" allows the access inside expressions. An optional parameter can limit the text to a fixed page.
„rawpage" is often used with the operators „tblsearch" and „tbllookup". See another chapter for an example.

Use of saved device modes

The configuration value (which also can contain calculation expressions) references a file, which was created with the tool "Devmode2File". If the driver version and the size of the device mode in the file are identical to that of the printer used in PrintMulti, the configuration values from the file are used instead of the standard device mode. The "device name" (shortened printer name to 30 characters) does not need to be the same. This allows the usage of the saved file with different printers having the same driver.

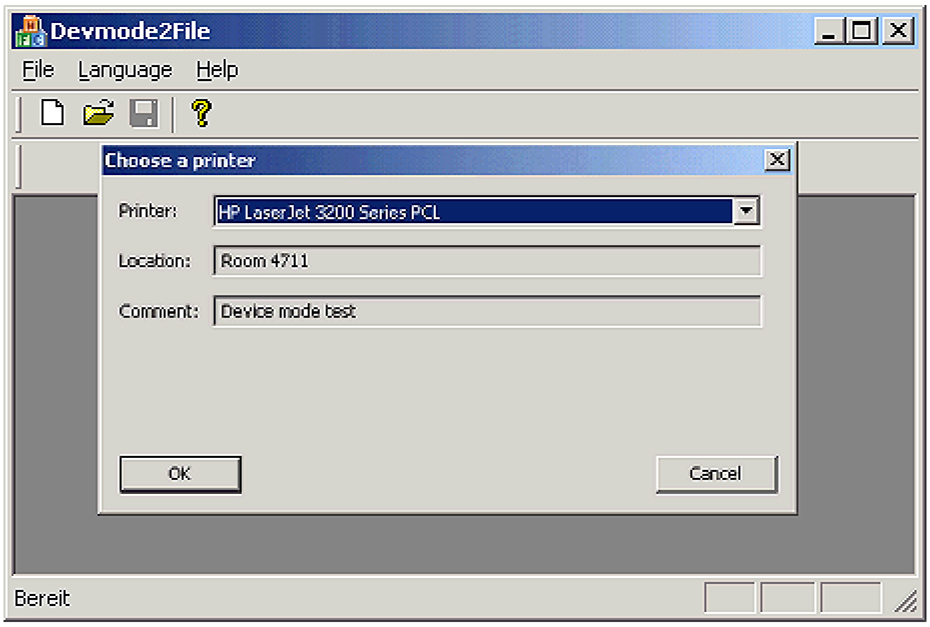
You can use saved device mode files to support driver specific things like output trays or water marks with PrintMulti.

The configuration applies always to all pages at the moment. A change for other page ranges will be implemented later on request. A workaround is to create multiple outputs with different page ranges using "FirstPage" and "LastPage" settings.

The following example shows the use of "Devmode2File" and the configuration settings for the task:

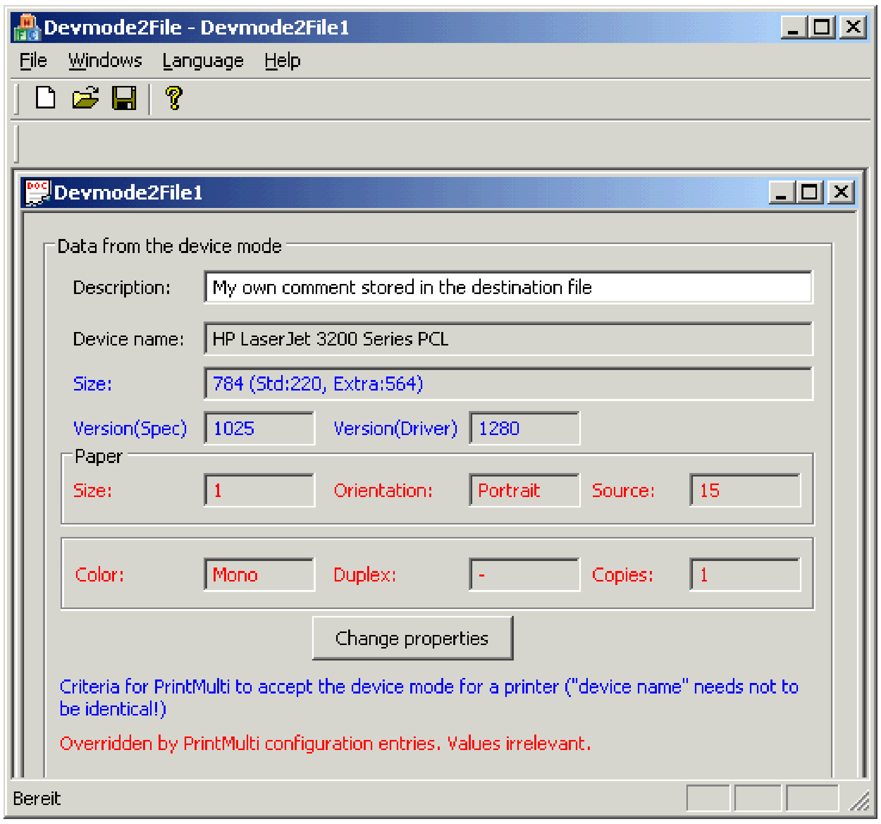
Set up a printer, which prints a job once normal and once with a water mark "copy". The printer driver (here HP Laserjet 3020 must support watermarks."

Start "Devmode2File", choose "New" and select the printer.



Printer: HP LaserJet 3200 Series PCL
Location: Room 4711
Comment: Device mode test

After "OK" the following dialog will pop up:



Data from the device mode

Description: My own comment stored in the destination file

Device name: HP LaserJet 3200 Series PCL

Size: 784 (Std:220, Extra:564)

Version(Spec) 1025 Version(Driver) 1280

Paper

Size: 1 Orientation: Portrait Source: 15

Color: Mono Duplex: - Copies: 1

Criteria for PrintMulti to accept the device mode for a printer ("device name" needs not to be identical!)

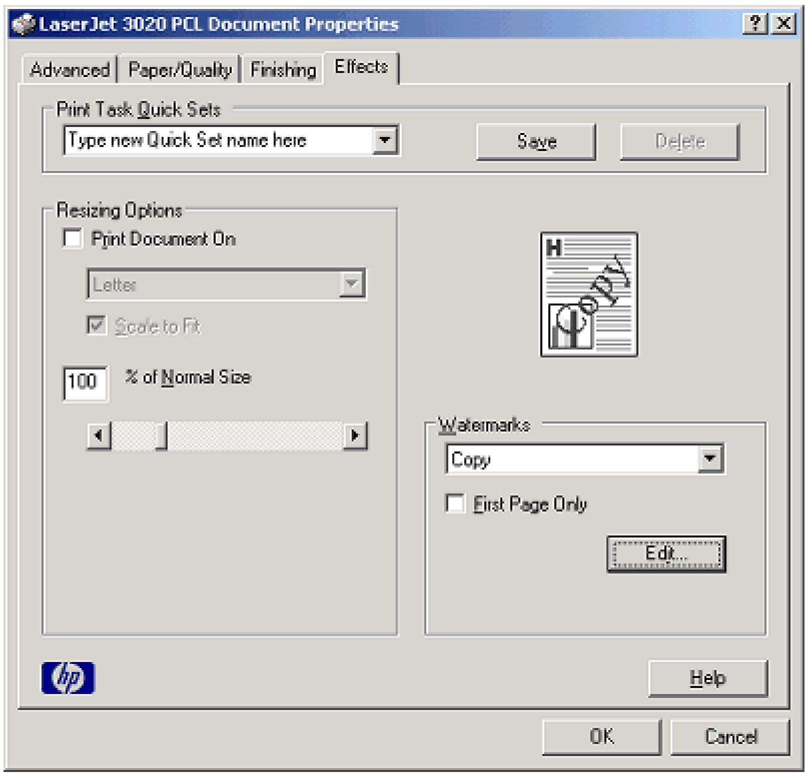
Overridden by PrintMulti configuration entries. Values irrelevant.

The "Description" is the only field that can be changed and could contain a description for this configuration.

The **blue** values are tested by PrintMulti if you choose a printer. They must be identical to the appropriate ones used with PrintMulti.

The **red** values are adapted if the properties has changed by using the printer settings dialog opened by the "Change Properties" button however they are overwritten by other PrintMulti settings and are therefore irrelevant.

The desired values (the output tray and the water mark) are driver specific. They are only displayed in the printer settings dialog. The following screen shots show the adaption of the water mark. Other changes can be performed similarily.


LaserJet 3020 PCL Document Properties

Advanced | Paper/Quality | Finishing | Effects

Print Task Quick Sets: Type new Quick Set name here — Save — Delete

Resizing Options
- [ ] Print Document On: Letter
- [✓] Scale to Fit
- 100 % of Normal Size

Watermarks: Copy
- [ ] First Page Only
- Edit...

Help — OK — Cancel

After the changes save the file with a suitable name ("e.g. "C:\program fles\PrintMulti\DeviceModi\HP3020_copy.dev ") and change the PrintMulti.ini accordingly:

```
[PrintWithCopy]
Active=1
ActionNormal=Print;ActionNormal
ActionCopy=Print;ActionCopy

[ActionNormal]
Printer=hp LaserJet 3020

[ActionCopy]
Printer=hp LaserJet 3020
Devmode1=%PM_INSTALLPATH%\DeviceModi\HP3020_copy.dev
```

Devmode2File assigns itself to the extension ‚*.dev", so that a double click on a dev-file opens the application automatically. Additionally ‚Drag & Drop" is possible.

Attention!

The name of the driver is not stored in a device mode. When loading the file only the sizes and versions can be tested (it is not desired to accept only files with the same printer name). It most cases not fitting files can be detected by the test. The use of a different driver can lead to a crash of the spooler. This is a driver issue and cannot catched by PrintMulti.

Please pay attention to use fitting devicemode files.

The ”Exec“ Action

With this action it is possible to run a program on the spool file itself. With the help of the SplViewer a preview for nearly all applications can be shown. With the viewer the print job can be sent to arbitrary printers with some options.

If the action is executed in a separated thread to free the printer for other jobs as soon as possible, the spool file will be copied.

PrintMulti waits until the process has finished. This is necessary to manage the spool files properly. If no separate thread is used, the printer is blocked until the calling application has finished (e.g. the viewer closed).

Here are the options in detail:

Action Exec			
Key	**Type**	**Default**	**Description**
Active	Bool	1	Use the action?
NewThread	Bool	False	Execution in a separate thread
ExecuteCmd ExecuteFlags ExecuteAsUser ExecuteShowWnd	String Int Bool Int	See action ”print“s	Settings for execution
RefreshConfig	**Bool**	**False**	**Rereads the configuration file after the current secion. Useful for dynamic changes to the configuration file**

Please note, the executed programs normally do not have access to network resources or to the registry path HKEY_CURRENT_USER.

If you copy the settings from

HKEY_CURRENT_USER\Printers

to

KEY_USER\.Default\Printer

can possibly avoid the problem.

Example:

```
[ActionPreview]
Active=1
ExecuteCmd=%ProgramFiles%\SplView\splview "#G"
; Flags for CreateProcess, 0x08000000 e.g. create no window for console app
ExecuteFlags=0
NewThread=y
; run as the printer user
ExecuteAsUser=y
; ShowCmd for ShowWindow (HIDE=0,Minimize=2,Maximize=3,...)
ExecuteShowWnd=3
```

The “SaveEmf“Action

The spool file accepted of PrintMulti consists of single pages in EMF format. The action allows writing each page to a seperate file for further processing.

The following options are available:

Hier die Einstellungen im Einzelnen:

Aktion Exec			
Einstellung	**Type**	**Default**	**Bedeutung**
Active	Bool	1	Use the action?
Destination	String		Path and base file name of the Emf file

Example:

```
[ActionSaveEmf]
Active=1
Destination=c:\temp\#P_#J
```

create files in the folder "c:\temp", which consists of the printer name, the job id and the page number formatted to five digits.

Using a printer "PrintMulti" with the first job id, the created files are:
PrintMulti_1_00001.emf
PrintMulti_1_00002.emf
...

PDF- and XPS creation with PrintMulti

This chapter explains three possibilities how to create PDF files automatically with PrintMulti.

The use of two-stage PDF printers

To this kind of PDF printers belong most freeware products (e.g. qvPDF, PDFCreator, FreePDF, eDocPrinter, Jaws PDF Creator and also Acrobat with its pdf printer/distiller pair).

All products in this category need a postscript printer to which the applications prints. The output is captured and passed through another application, which converts the postscript file to PDF. ”Conversion programs“ are ”Ghostscript“, the ”Acrobat Distiller“ or own tools.

What all products have in common is that the printing of documents to a file leads to a postscript and not to a PDF file. This behavior can also be observed with other file printing capable programs like Word. It is pointless to use the ”Save2File“ option in PrintMulti, because the resulting file only contains postscript code. The solution is to involve a script with ”ExecuteCmd“ (see third section ”direct creation with PrintMulti“) or to use the functions of the particular printers. In this case the name of the pdf file cannot be modified by PrintMulti.

Printers from this category are often useful if the extended functions which operate with PDF files can be used (e.g. E-Mail sending … - qvPDF offers many additional features).

The example below shows a configuration for the PDF printer ”qvPDF“ and the physical printer ”Laserjet“. PDF output options must be set inside qvPDF.

```
[PDFLaserjet]
Active=1
PrintSelf=0
ActionPDF=Print;PDF
ActionPrint=Print;Laser

[PDF]
Active=1
Printer=qvPDF

[Laser]
Active=1
Printer=Laserjet
```

The use of ”native“ PDF/XPS printers

There are PDF printers, which are capable of creating PDF files in one step. These work with PrintMulti as well as with Word or other applications.

We advise the ”Amyuni PDF Converter“ in this category (since we use it for ourselves). There are certainly other products which are designed similar and work well.

The ”Microsoft XPS Document Writer“, which is also a ”native“ driver, can be used to create XPS files.

The name of the output file can be configured with the ”Save2File“ option.

The example shows the creation of a PDF and a XPS file with concurrent printing to the printer ”Laserjet“.

```
[PDFXPSLaserjet]
Active=1
PrintSelf=0
ActionPDF=Print;PDF
ActionXPS=Print;XPS
ActionPrint=Print;Laser

[PDF]
Active=1
Save2File=D:\PDFFiles\#P\#(%Y-%m)S\#K_#C.PDF
Printer=Amyuni PDF Converter

[XPS]
Active=1
Save2File=D:\XPSFiles\#P\#(%Y-%m)S\#K_#C.XPS
Printer=Microsoft XPS Document Writer

[Laser]
Active=1
Printer=Laserjet
```

Printing a test page with an actual ID of 1 for the printer will create the following files:

”D:\PDFFiles\PDFXPSLaserjet\2008-06\Testseite_1.pdf“ and

“D:\XPSFies\ PDFXPSLaserjet\2008-06\Testseite_1.xps“

Direct creation with PrintMulti (Ghostscript)

You need an installed Ghostscript. If you plan to create PDF/A compatible documents, Ghostscript starting from version 8.61 should be used. Tests with the pdfaPilot from Callas Software confirm the full PDF/A compatibility (which is not reached in all cases by Acrobat 8.0!). The pilot can also be used to convert not PDF/A compatible documents to the desired format.

Additionally a suitable postscript printer driver is needed. Others options included ”Ghostscript PDF“ (”ghostpdf.inf“ in the lib-folder) or the ”Rumborak PDF-Writer Plus“ (http://www.rumborak.de/produktives).

The drivers allow inserting distiller commands into the postscript data stream to influence the following PDF creation.

The call of Ghostscript is done by the included VBScript file “gscreatepdf1.vbs“ (or ”gscreatepdf.vbs“ in earlier versions). The files should be used as examples for your own extensions, especially if you need to further process the PDF file. The calling script is set with the ”ExecuteCmd“ option.

PrintMulti offers assistance to access the printer relevant information from inside the batch file.

1. Command line parameters, which may contain macros (e.g. #G).
2. Environment variables, which are set automatically (see addendum, e.g. ”PM_PRINTER“)
3. Environment variables, which are set in the printer section of the ”PrintMulti.ini“. They must start with an exclamation mark (e.g.: !GS_PDFMODE=PDF/A)

E-Mail sending ability was added with the version 1.0.3.0. More information can be found in the example section.

A full PDF/A compatibility seams to be only reached, if the following options are set for the used postscript printer:

TrueType-Font:"Download as Softfont"

TrueType-Font Download Option:"Outline"

Network problems

A print processor is a DLL, which is loaded by the spooler process ”spoolsv.exe“ or for active driver isolation (Windows 7 and Server 2008 R2) by the process "printisolationhost.exe". Both processes run with the system account.

The system account has many rights, but also some restrictions:

1. Uses the standard user profile (HKEY_USERS\.DEFAULT). It could be necessary to copy settings from the user registry path.
2. Access to network resources is only possible, if “null sessions” are allowed. Further information t: http://support.microsoft.com/kb/289655

No access to network printers:

If the access does not work, you have three possibilities.

1. Install the printer locally. As new ”Local Port“ use the printer share on the server with UNC-names.
2. Copy data from
   ”HKEY_CURRENT_USER\Printers\Connections“
   ”HKEY_CURRENT_USER\Printers\DevModePerUser“
   ”HKEY_CURRENT_USER\Printers\DevModes2“
   to
   ”HKEY_USERS\.DEFAULT“
   in the appropriate path. This does not always work!
3. Experiment with the settings ”UseSystemAccount“ in the printer section and try to allow the null session access.

Examples

Examples for calculable expressions

Printing dependent on the number of pages

It may be useful to output print jobs with many pages to more powerful printers. Under the assumption that the printer is named ”CommonPrinter“ for the users, the powerful printer ”FastPrinter“ and a slow one ”SlowPrinter“ the configuration would look this (the page limit to switch printers is 100 pages):

```
[Common]
; 32 show only calculation messages. 59 all messages without job messages
LogMask=32
…
[CommonPrinter]
Active=1
ActionFast=Print;Fast;$(#Z;100;>=)
ActionSlow=Print;Slow;$(#Z;100;<)

[Fast]
Printer=FastPrinter

[Slow]
Printer=SlowPrinter
```

Here is another possibility:

```
[CommonPrinter]
Active=1
ActionSlowFast=Print;SlowFast

[SlowFast]
Printer=$(#Z;100;>=;FastPrinter;SlowPrinter;?)
```

The important part of the log file would look like this:

```
CALC;...;print ;SlowFast;SlowFast.Printer: 1 >= 100 ->0
CALC;...;print ;SlowFast;SlowFast.Printer: (0)? FastPrinter : SlowPrinter ->SlowPrinter
```

User dependent printing

Some users may have special rights. The ”contains“ operator is useful to configurate this kind of problem. In the example one some people may print in color.

```
[Common]
_ColorUsers=|dieter|gustav|bärbel|
_IsColorUser=$(#(_ColorUsers)I;upper;|;#U;+;|;+;upper;contains)

[CommonPrinter]
Active=1
ActionColor=Print;Color

[Color]
Printer=ColorPrinter
Color=$(#(Common._IsColorUser)I;1;0;?)
```

Part of the log files shows the calculation of the complicated expression ”_IsColorUser“

```
CALC;...;Read entry #(Common._IsColorUser)I
CALC;...;Read entry #(Common._ColorUsers)I
```

```
CALC;...;Common._IsColorUser: Upper("|dieter|gustav|bärbel|")->
"|DIETER|GUSTAV|BÄRBEL|" Locale:German
CALC;...;Common._IsColorUser: "|" + "dieter" ->"|dieter "
CALC;...;Common._IsColorUser: "|dieter" + "|" ->"|dieter|"
CALC;...;Common._IsColorUser: Upper("|dieter|")->"|DIETER|" Locale:German
CALC;...;Common._IsColorUser: "|DIETER|GUSTAV|BÄRBEL|" contains "|DIETER|" ->1
CALC;...;Color.Color: (1)? 1 : 0 ->1
```

The example shows the following functions:

Use of the macro `#(…)I` to reference other sections. The underline before the name is just for clearness (in opposite to a leading ”!“ which marks variable as environment variable for script executing).

How a string compare can be done independent from the capitalization. No Windows regional settings will be used here (in opposite to the compare operators like ”<“). From the log file can be determined, that ”German“ local setting is used.

Expressions are always calculated new with the current context.

The last line could also be written as ”`Color=#(Common._IsColorUser)I`“, since color only accept the values 0 or 1. The assignment with the conditional operation can easily be transferred similarly to other settings.

The log file is important to understand complicated expressions.

Document dependent printing

You might want to set an archive name which is dependent on the printed document for concurrent archiving while printing. You will need that name in the script file where the archive import file is created.

The example uses the Amyuni PDF Converter for creating the PDF file. The called script should delete the file after importing it. Only the involved parameters are showed here.

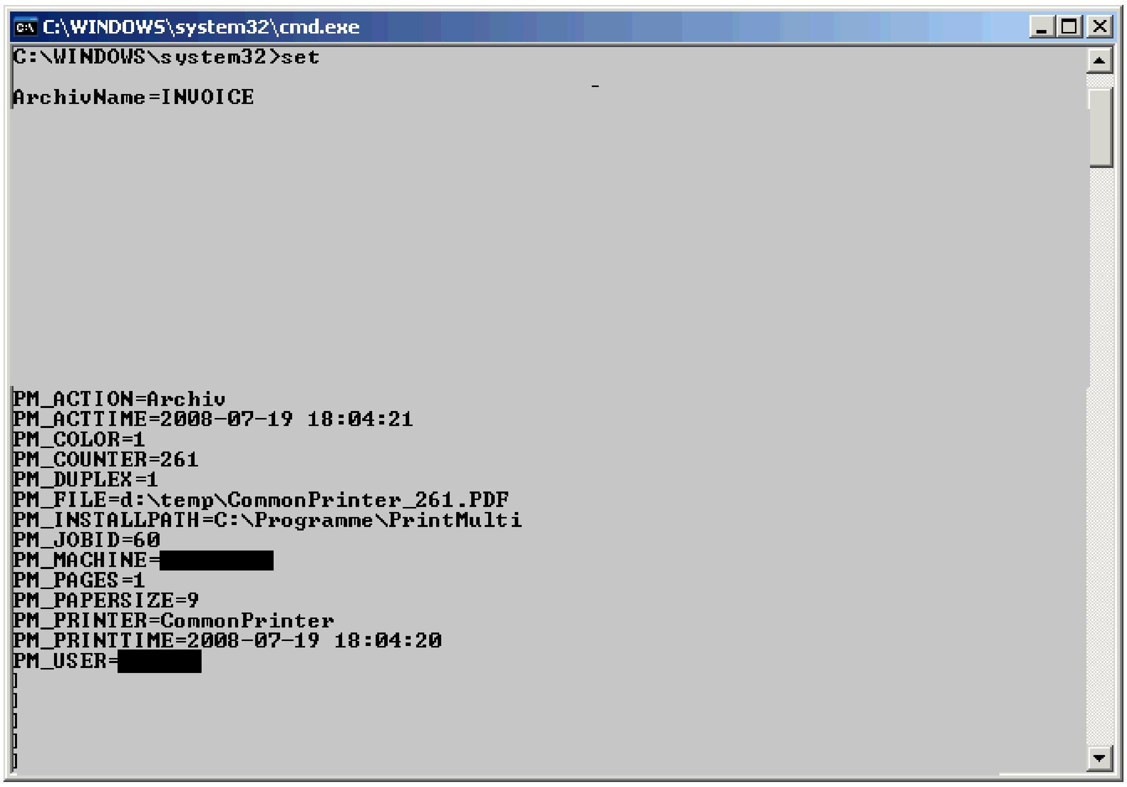
```
[CommonPrinter]
Active=1
ActionArchiv=Print;Archive

[Archive]
Printer=Amyuni PDF Converter
Save2File=%TEMP%\#P_#C.PDF
_IsInvoice=$(#D;upper;INVOICE;contains)
_IsDelivery=$(#D;upper;DELIVERY;contains)
!ArchivName=$(#(_IsInvoice)I;INVOICE;#(_IsDelivery)I;DELIVERY;UNKNOWN;?;?)
Execute=y
ExecuteCmd=cmd /K echo #G
```

```
CALC;...;Read entry #(Archiv._IsInvoice)I
CALC;...;Archiv._IsInvoice: Upper("Microsoft Word - MyInvoice.doc")->"MICROSOFT WORD -
MYINVOICE.DOC" Locale:German
CALC;...;Archiv._IsInvoice: "MICROSOFT WORD - MYINVOICE.DOC" contains "INVOICE" ->1
CALC;...;Read entry #(Archiv._IsDelivery)I
CALC;...;Archiv._IsDelivery: Upper("Microsoft Word - MyInvoice.doc")->"MICROSOFT WORD -
MYINVOICE.DOC" Locale:German
CALC;...;Archiv._IsDelivery: "MICROSOFT WORD - MYINVOICE.DOC" contains "DELIVERY" ->0
CALC;...;Archiv.!ArchivName: (0)? DELIVERY : UNKNOWN ->UNKNOWN
CALC;...;Archiv.!ArchivName: (1)? INVOICE : UNKNOWN ->INVOICE
```

The example opens a command window.

Remember: All variables starting with a ”!“ in a section are set as environment variable with the calling process (without the ”!“). It is also possible to transfer the archive name to the script by a command line parameter. This would look ”… #G #(!ArchivName)I“

.



```
C:\WINDOWS\system32>set

ArchivName=INVOICE

PM_ACTION=Archiv
PM_ACTTIME=2008-07-19 18:04:21
PM_COLOR=1
PM_COUNTER=261
PM_DUPLEX=1
PM_FILE=d:\temp\CommonPrinter_261.PDF
PM_INSTALLPATH=C:\Programme\PrintMulti
PM_JOBID=60
PM_MACHINE=[redacted]
PM_PAGES=1
PM_PAPERSIZE=9
PM_PRINTER=CommonPrinter
PM_PRINTTIME=2008-07-19 18:04:20
PM_USER=[redacted]
```

Some internal details are grayed out. The created environment variable are good visible.

The example could also be configured more easily (Difference: If none of the conditions is fulfilled, no action is called):

```
[CommonPrinter]
Active=1
ActionArchiv1=Print;InvoiceArchiv;$(#D;upper;INVOICE;contains)
ActionArchiv2=Print;DeliveryArchiv;$(#D;upper;DELIVERY;contains)

[InvoiceArchiv]
Printer=Amyuni PDF Converter
Save2File=%TEMP%\#P_#C.PDF
!ArchivName=INVOICE
Execute=y
ExecuteCmd=cmd /K echo #G

[DeliveryArchiv]
Printer=Amyuni PDF Converter
Save2File=%TEMP%\#P_#C.PDF
!ArchivName=DELIVERY
Execute=y
ExecuteCmd=cmd /K echo #G
```

The same result is achieved by moving the condition after the Action-Entries to the flag ”Active=“ inside the sections.

```
[CommonPrinter]
Active=1
ActionArchiv1=Print;InvoiceArchiv
ActionArchiv2=Print;DeliveryArchiv

[InvoiceArchiv]
Active=$(#D;upper;INVOICE;contains)
...
```

Date or time dependent printing

You might want to use different printers between eight in the evening until six in the morning.

```
[CommonPrinter]
Active=1
ActionNight=Print;NightPrint;$(#(%H)S;20;>=;#(%H)S;6;<=;or)
ActionDay=Print;DayPrint;$(#(%H)S;20;<;#(%H)S;6;<;and)

[NightPrint]
Printer=SilentPrinter

[DayPrint]
Printer=LoudPrinter
```

The log file would contain the following entries if the printing was between 21 and 22 o“ clock.

```
CALC    ;…;CommonPrinter.ActionNight: 21 >= 20 ->1
CALC    ;…;CommonPrinter.ActionNight: 21 <= 6 ->0
CALC    ;…;CommonPrinter.ActionNight: 1 OR 0 ->1
DEBUG   ;…;print ;NightPrint          ;Start Action:“NightPrint“
```

Use of the document content

The new commands and options “tblsearch”, “tbllookup”, “rawpage”, “writetexttofile” offer the possibility to use information from the printed document.

The “tbllookup” operator allows a simple lookup for a statement in a configuration section. The example shows how to simplify the “user dependend printing”.

The “tblsearch” operator picks up every key from the section and returns the (calculated) value on the right side, if the key is contained in the corresponding parameter (“1;rawpage” in the example).

Attention!!! “tblookup” is case insensitive; “tblsearch” case sensitive.

```
 [ColorUsers]
Dieter=$(#Z;100;<)
Gustav=0

[Printers]
#invoice#=PrinterInvoice
#delivery#=PrinterDelivery

[ActionTblTest]
Active=1
Color=$(;ColorUsers;#U;0;tbllookup)
Printer=$(;Printers;1;rawpage;TestPrinter;tblsearch)
; Example for a different configuration file and the use of all pages
; Printer=$(%PM_INSTALLPATH%\users.ini;;rawpage;TestPrinter;tblsearch)
_TextFile=%TEMP%\#P_#C.txt
WriteTextToFile=#(_TextFile)I
Execute=y
ExecuteCmd=wscript "%PM_INSTALLPATH%\PDFCreation\textextract.vbs" "#(_TextFile)I"
```

Some comments:

The right side of an assignment may contain calculations. „Dieter” prints only in color if the number of pages for a document is less than 100 pages.

If the file parameter (the first one, which is empty in the example) contains a reference to a file, this file will be reread for each access of the operator. This allows to dynamic change if needed.

The new “rawpage” operator supplies a text, which contains all printed statements without any delimiter. The optional first parameter may contain a page number starting with one. If not configured, all pages are contained, delimited by a CR/LF between the pages. To identify a specific text it should be surrounded by special characters (here ‘#’). The text can be invisible (or very small) by the use white font color. Normally it must be inside the printed area.

“WriteTextToFile” is useful to solve problems, if something is not found as expected. The file can also be examined by a script like in the example. The installation contains a sample script “textextract.vbs”. The format of the created text is always “UTF16”.

Logging of print jobs

Let us assume, you want to create a log entry in a database for each print job, which user has printed it on which printer and how many pages in color or black white the job had. You can examine the log files, but in this example we use a program to perform that.

Install a printer with name ”SumPages“ and assign ”PrintMulti“ as print processor.

The Ini-Section looks like this:

```
[SumPages]
Active=1
Action=Print;SumPages
Execute=yes
ExecuteFlags=0x08000000
ExecuteCmd=%windir%\sumpages.bat ”#U“ ”#P“ #c #Z #z
```

”sumpages.bat“ would contain code to insert the parameters into a database table. Here I use a simple batchfile, which appends the parameters to a CSV-file.

```
echo %1,%2,%3,%4,%5 >>%windir%\pages.csv
```

The destination file contains for every print job one line with the user, the printer, 1 for color / 0 for black white, the number of pages and the number of copies.

Some further comments:

The action points to itself – acts as a main entry and action entry. A direct recursive call of PrintMulti will be suppressed internally. To use this construct is a little dangerous und could lead to an endless loop. So be carefully or better avoid it.

Since no ”printer“-entry was set, the name of the section ”SumPages“ will be used as printer name.

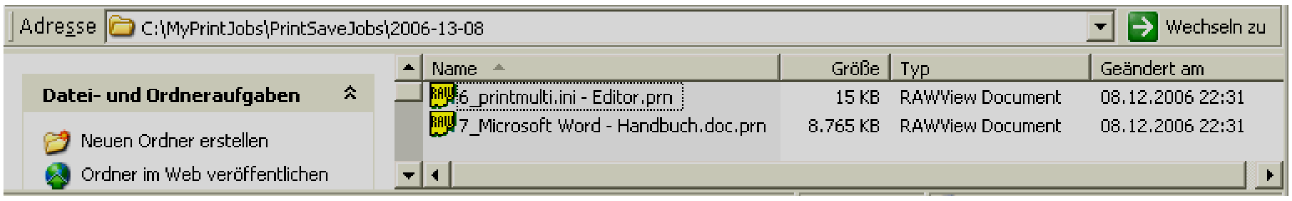
After the printing the command will be execute. The flags cause no console window to pop up.

Saving of print jobs

You may wish to save all print jobs, so that they can later be reprinted or send to third party companies to print them.

```
[PrintSaveJobs]
Active=1
Action1=Print;PrintJobs
Action2=Print;SaveJobs

; print the jobs
[PrintJobs]
Active=1
Printer=RealPrinter

; save the jobs and create a log file
[SaveJobs]
Active=1
Printer=RealPrinter

; Force one copy
TotalCopies=1
DrvCopies=1
Save2File=c:\MyPrintJobs\#P\#(%Y-%m-%d)S\#C_#K.prn
Append2File=0
Execute=yes
ExecuteFlags=0x08000000
ExecuteCmd=%windir%\savepages.bat
```

The files would be saved like this:


Adresse C:\MyPrintJobs\PrintSaveJobs\2006-13-08 Wechseln zu

Datei- und Ordneraufgaben
Neuen Ordner erstellen
Ordner im Web veröffentlichen

Name	Größe	Typ	Geändert am
6_printmulti.ini - Editor.prn	15 KB	RAWView Document	08.12.2006 22:31
7_Microsoft Word - Handbuch.doc.prn	8.765 KB	RAWView Document	08.12.2006 22:31

The batchfile ”savepages.bat“ contains only one line:

```
echo ”%PM_PRINTTIME%“;“%PM_USER%“;“%PM_PRINTER%“;“%PM_FILE%“ >>c:\MyPrintJobs\jobs.csv
```

After the two print jobs, the job file would contain:

```
”2006-12-08 22:31:29“;“dieter“;“PrintSaveJobs“;“c:\MyPrintJobs\PrintSaveJobs\2006-13-08\6_printmulti.ini - Editor.prn“
”2006-12-08 22:31:52“;“dieter“;“PrintSaveJobs“;“c:\MyPrintJobs\PrintSaveJobs\2006-13-08\7_Microsoft Word - Handbuch.doc.prn“
```

The example shows the usage of the environment variable, which are listened in the addendum.

By the way, ”#D“ in the file name could give some problems. If you print for example a document from an editor, the document name could contain the full path of the printed file. If you use this in a ”Save2File“-line it will lead to an invalid filename. Please use ”#K“ instead of ”#D“ in this case.

Now every day a new directory will be created and all jobs, that are printed to the printer ”PrintSaveJobs“ were stored there. The CSV-File contains references and information about the file and could be analyzed.

The data will be appended to the file, if ”Append2File“ is set to one. This could be useful if you give the data to another person to print more jobs at once.

”#C“ is a unique job counter for the printer ”PrintSaveJobs“ in the example, which is increased with every print job. It is stored in the following key and must be reset manually.

”*HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Printers\PrintSaveJobs\PrinterDriverData\JobCounter*“

Working with PDF files

The sub folder "PDFCreation" of the PrintMulti installation includes the script "gscreatepdf1.vbs". It was extended to send PDF files as attachment to destinations contained in the document.

The base function of the script is the creation of PDF files (or PDF/A) with the help of Ghostscript (=GS) and to show some ideas what can be done with them. An existing GS installation, which should be automatically detected, is needed.

Many configuration options are defined by fixed entries in the PrintMulti.ini. They could also be extracted from the printed text like some of the other options.

Some points about the PDF creation:

```
[ActionCreatePDFScript]
Printer=Ghostscript PDF
Save2File=%TEMP%\#P_#C.ps
Color=1
Execute=yes
ExecuteFlags=0x08000000
ExecuteCmd=cscript "%PM_INSTALLPATH%\PDFCreation\gscreatepdf1.vbs" "#G"
 ►"%TEMP%\#K_#C_#U.pdf"
!GS_COMPATIBILITY=1.4
; screen,ebook,printer,prepress,default
!GS_PDFSETTINGS=screen
; a0,a1,..,a10,c0,..,c6,11x17,ledger,legal,letter,... uncomment for fixed paper
 ►size
;!GS_FIXPAPERSIZE=a0
; Additional gswin32 parameter. See use.htm in gs\doc-path
; Can also be appended to the command line below.
;!GS_EXTRAPARAMS=-dPDFWRDEBUG
; Append log messages to that file. Comment out for no debug messages
!GS_LOGFILE=%TEMP%\out.txt
; Set some internal parameters for PDF/A compatibility and use PDFA_def.ps file
!GS_PDFMODE=PDF/A
```

You need a postscript printer. The printer in the example "Ghostscript PDF" is included in the GS installation (inf-file in the lib folder).
The postscript file will be saved in the temporary folder under a unique name ("#C" is a counter). The destination folder is normally "C:\Windows\Temp"-, since the script is called in the context of the system account and no user registry hive is available.

The script receives the postscript file (#G) and the destination PDF file as parameter.
For debug output the "ExecuteFlags" must be set to 0 and the cscript be called with "ExecuteCmd=cmd /k cscript ..."

All variable in the section that starts with an exclamation mark are set as environment variable for the executed program.

If you set !GS_LOGFILE the file contains debug output from the GS call. That is useful for error detection. In some cases GS is not able to convert the postscript file successfully.

PDF/A output require a colour profile. The PrintMulti installation stores one in the correct Windows folder (see installation hints at the beginning of the document).

Below is a section from the script which shows how the command line to call GS is build from the different parameters:

```
compatibility = procEnv("GS_COMPATIBILITY")
pdfsetting    = procEnv("GS_PDFSETTINGS")
fixpapersize  = procEnv("GS_FIXPAPERSIZE")
debugoutput   = procEnv("GS_LOGFILE")
extraparams   = procEnv("GS_EXTRAPARAMS")
pdfmode       = procEnv("GS_PDFMODE")

'Build command line file for ghostscript
```

```
paraFileName=createGuid()+".txt"
Set fso = CreateObject("Scripting.FileSystemObject")

'Base cmd line
cmdLine = "-q -dSAFER -dNOPAUSE -dBATCH -sDEVICE=pdfwrite "

'PDFMODE has priority over some other settings
if ucase(pdfmode) = "PDF/A" Then
  pdfaxFile = createGuid()+".ps"
  ReplaceEnvironment "%PM_INSTALLPATH%\PDFCreation\PDFA_def.ps", pdfaxFile
  cmdLine = cmdLine + "-dPDFA -dUseCIEColor -sProcessColorModel=DeviceCMYK "
Else
  if compatibility <> "" Then cmdLine = cmdLine +  "-
      CompatibilityLevel="+compatibility+" "
  if pdfsetting    <> "" Then cmdLine = cmdLine + "-dPDFSetting=/"+pdfsetting+" "
  if fixpapersize  <> "" Then cmdLine = cmdLine + "-sPAPERSIZE="+fixpapersize+" "
  if extraparams   <> "" Then cmdLine = cmdLine + extraparams + " "
End if
```

Sending of E-Mails:

The powerful tool "blat.exe" is used for sending the mail (www.blat.net). There are certainly other command line tools for sending mails with the possibility to attach files.

```
; to test mail sending, uncomment the three following lines
!GS_SENDMAIL=1
_DstFile=%TEMP%\#P_#C.txt
WriteTextToFile=$(#(!GS_SENDMAIL)I;1;=;#(_DstFile)I;;?)

; Options used to send the printed file as attached pdf file to an address
; defined in the example document
!GS_MAILFILE=%PM_INSTALLPATH%\PDFCreation\mailbody.txt
; activate for a log file which shows the output of blat, which is used for
; mail sending
!GS_MAILLOGFILE=%TEMP%\mail.log

;server and login data
!GS_MAILSERVER=xxxxxx
!GS_MAILSERVERUSER=xxxxxxx
!GS_MAILSERVERPASSWORD=xxxxxxxx
!GS_MAILREPLYADDRESS=info@lvbprint.de

; set the name to be used in the script
!GS_TEXTFILE=#(_DstFile)I
```

The server and login account data are configured in the PrintMulti.ini.

Destination address and subject are extracted from the word document (see SendMailExample.doc).

The body text is taken from the fixed file "mailbody.txt" in the example.

A log file is written, if "!GS_MAILLOGFILE" is defined.

The switch "!GS_SENDMAIL" controls if a mail should be sent at all. It is also examined in the script.
If the switch is set, the unformatted page text is written to a file in the temp folder with the help of the command "WriteTextToFile".
In the other case the expression is evaluated to an empty string and ignored (see the log file).

The name of the text file is assigned to the environment variable "!GS_TEXTFILE".

The script deletes the text file, so that no unnecessary files remain.

The extraction of the data from the source text is done by the function "ExtractDataFromText" inside the script.

Here is the part of the script which handles mail sending.

```
' must be set in the printmulti.ini
sendMail = procEnv("GS_SENDMAIL")
```

```
' needs blat.exe in the PDFCreation folder
' needs CmdRedirect.exe in the PDFCreation folder (set above)
' adjust log file, user name and password
If sendMail = "1" Then
	Dim blatExeFile, mailfile, subject, address, tempMailLogFile, mailLogFile
	Dim mailserver, mailserveruser, mailserverpassword, mailreplyaddress, strContent

	' get the name of the text file
	textFileName = procEnv("GS_TEXTFILE")

	' load the complete text into a variable
	set textFile = fso.OpenTextFile(textFileName, ForReading, false, TriStateTrue)
	strContent=textFile.ReadAll()
	textFile.Close

	' get the address and the subject from the printed text
	address = ExtractDataFromText(strContent, "GS_MAILSENDERADDRESS")
	subject = ExtractDataFromText(strContent, "GS_MAILSUBJECT")

	 blatExeFile=shell.ExpandEnvironmentStrings("%PM_INSTALLPATH%\PDFCreation\blat.exe")
	' the rest is defined as constants in the PrintMulti.ini file
	mailfile = procEnv("GS_MAILFILE")
	mailLogFile = procEnv("GS_MAILLOGFILE")
	mailserver = procEnv("GS_MAILSERVER")
	mailserveruser = procEnv("GS_MAILSERVERUSER")
	mailserverpassword = procEnv("GS_MAILSERVERPASSWORD")
	mailreplyaddress = procEnv("GS_MAILREPLYADDRESS")

	gsCmdLine = ...

	shell.Run(gsCmdLine),0,true

	If mailLogFile <> "" Then
		AppendFile2File mailLogFile, tempMailLogFile, 2
		Wscript.echo mailLogFile + ":Append:"+tempMailLogFile
		fso.DeleteFile tempMailLogFile
	End If

	fso.DeleteFile textFileName
End if
```

Addendum

Environment variables set, when executing programs

PM_ACTION	Name of the action (#A)
PM_COUNTER	Unique counter (#C)
PM_JOBID	JobId from Windows (#J)
PM_USER	Username (#U)
PM_MACHINE	Computer name of the printing computer (#M)
PM_PRINTER	Name of the main printer (#P)
PM_PAGES	Number of pages (#Z)
PM_COLOR	Color printing ? (#c)
PM_PAPERSIZE	Paper size (#s)
PM_DUPLEX	Simplex, Duplex Horizontal or Simplex Vertical (#d)
PM_ACTTIME	Actual time (#T)
PM_PRINTTIME	Time from the print job (#S)
PM_FILE	Output file (#G)

Paper formats

The following paper formats are recognized. The text is replaced with the number in the first column

Value	**Paper format description**
1	Letter
2	Letter Small
3	Tabloid
4	Ledger
5	Legal
6	Statement
7	Executive
8	A3
9	A4
11	A5
12	B4 (JIS)
13	B5 (JIS)
14	Folio
15	Quarto
16	10x14
17	11x17
18	Note
19	Envelope #9
20	Envelope #10
21	Envelope #11
22	Envelope #12
23	Envelope #14
24	C size sheet
25	D size sheet
26	E size sheet
27	Envelope DL
28	Envelope C5
29	Envelope C3
30	Envelope C4
31	Envelope C6
32	Envelope C65
33	Envelope B4
34	Envelope B5
35	Envelope B6
37	Envelope Monarch
27	DL
28	C5
29	C3
30	C4
31	C6
32	C65
33	B4
34	B5
35	B6
37	Monarch

Values for ”ExecuteShowWnd“

You have to use the numbers

SW_HIDE	0
SW_SHOWNORMAL	1
SW_NORMAL	1
SW_SHOWMINIMIZED	2
SW_SHOWMAXIMIZED	3
SW_MAXIMIZE	3
SW_SHOWNOACTIVATE	4
SW_SHOW	5
SW_MINIMIZE	6
SW_SHOWMINNOACTIVE	7
SW_SHOWNA	8
SW_RESTORE	9
SW_SHOWDEFAULT	10
SW_FORCEMINIMIZE	11