

printer

Brian Gontowski

Copyright © Copyright 1994-1995 by Brian Gontowski.

COLLABORATORS

	<i>TITLE :</i> printer		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Brian Gontowski	August 23, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 printer	1
1.1 V42 Printer Subsystem	1
1.2 V42 Printer Subsystem - Features	1
1.3 V42 Printer Subsystem - Implementation	2
1.4 V42 Printer Subsystem - Compatability	3
1.5 V42 Printer Subsystem - OS Requirements	3
1.6 V42 Printer Subsystem - Open Issues	4
1.7 V42 Printer Subsystem - Printer Manager	4
1.8 V42 Printer Subsystem - Preferences	5
1.9 V42 Printer Subsystem - PRT: Handler	6
1.10 V42 Printer Subsystem - Includes and Autodocs	7

Chapter 1

printer

1.1 V42 Printer Subsystem

V42 Printer Subsystem - Written by Brian Gontowski.

This archive is Copyright © 1994-1995 by Brian Gontowski.
It may be distributed AS IS for non-profit purposes.
Please e-mail me for information on distribution with
a commercial product.

This software requires AmigaOS 3.0 (V39) or higher.

Features

Implementation

Compatability

OS Requirements

Open Issues

Printer Manager

Preferences

PRT: Handler

Includes and Autodocs

E-MAIL ADDRESSES:

blg@cherry-semi.com

bgontowski@bix.com

1.2 V42 Printer Subsystem - Features

Multi-printer support for up to 25 printer running at the same time.
Multi-threaded printing for devices that support it (i.e. network).

Queuing for devices that don't support multi-threading (i.e. parallel port).
Printing to file using an asl.library file requester.
UNIT_PRINTERMANAGER for controlling print jobs.

1.3 V42 Printer Subsystem - Implementation

PrinterData

The structure is part of the device base. Therefore, for multiple printers to work, this device base is duplicated and the copy filled in for each print job (a "print job" is an OpenDevice, possible write, and a CloseDevice)

Some of the fields were reused for the new printer.device. pd_Task and pd_OldStk are no longer needed and were private structure outside of the printer.device. These were filled in with other private or read-only variables.

OpenDevice()

The first open device creates a "printer.manager" process. This process takes care of managing the print jobs.

After this, a separate process is created to handle the actual print job. This only happens if the unit requested is not UNIT_PRINTERMANAGER.

"printer.device" process

On start up, Preferences is loaded from printer.prefs/printergfx.prefs (for unit 0) or printer_UNIT.prefs/printergfx_UNIT.prefs (for units 1-24), the printer driver is loaded and initialized.

NOTE: printergfx.prefs is only used if a PGFX chunk is not found in printer.prefs. Also, unit 0 will use defaults if no preferences files are found.

If the unit for the opening printer is 0, the \$DefaultPrinter variable is set to a different unit, and the flag PRDF_NODEFAULT is not set, the printer will map to the other unit. This allows "PrtMgr" to control a default unit (since most programs will only open unit 0).

This process registers with the "printer.manager" process and respects flags such as stop and abort.

The actual output device does not get an open attempt until the first write. The reason for this is some program open the printer device to find out page widths, etc.

At the point of the first write, an open is tried for the output device. If it fails because the output device is busy, it is marked as a single-treaded output device. For them on, this process checks with the printer.manager to see if it is alright to open the output device (this depends on the order in the queue).

"printer.manager" process

This process provides the following information...

- Stopping/Starting/Moving/Aborting requests
- Providing lists for listener programs (such as PrtMgr)

- Controlling which print job is next in the queue

1.4 V42 Printer Subsystem - Compatability

The basic functionality from the application remains the same. Either `printer.device` or `PRT:` can be used to print. Programs that "hog" the printer by keeping it open during the entire programs will remain "hogs" with this version of the `printer.device`. Other request will just get queued for single-threaded devices. Multi-threaded devices (such as printing to a file) will print regardless of the "hog".

Since all old programs use `printer.device` unit 0, the `$DefaultPrinter` variable allows use of several printers with minimum user interaction.

1.5 V42 Printer Subsystem - OS Requirements

```
printer.device
  exec.library (V37)
  dos.library (V37)
  intuition.library (V37)
  graphics.library (V37)
  utility.library (V37)
  locale.library (V38 - optional)
  iffparse.library (V37)
  asl.library (V38 - only required for "print to file")
  timer.device (V37)
```

```
Printer Manager
  exec.library (V39)
  dos.library (V39)
  intuition.library (V39)
  graphics.library (V39)
  gadtools.library (V39)
  utility.library (V39)
  workbench.library (V39)
  icon.library (V39)
  locale.library (V38)
  asl.library (V38)
  iffparse.library (V39)
  datatypes.library (V40 - optional)
  commodities.library (V38)
  printer.device (V42)
```

```
Printer Preferences
  exec.library (V39)
  dos.library (V39)
  intuition.library (V39)
  graphics.library (V39)
  gadtools.library (V39)
  utility.library (V39)
  workbench.library (V39)
  icon.library (V39)
```

```
locale.library (V38)
asl.library (V38)
iffparse.library (V39)
```

```
printer-handler
  exec.library (V37)
  dos.library (V37)
  utility.library (V37)
  iffparse.library (V37)
  printer.device (V42)
```

The new printer.device can run without the program PrtMgr.
So, V37 is the minimum for using this printer.device (without file support).

1.6 V42 Printer Subsystem - Open Issues

printer.device

The current V42 printer.device has the drawback that it stores each ioreq in memory before dumping to the output device. This means that a rastport dump will be stored entirely in memory before being outputted. This needs to change. At the same time, the concept of printer spooling would probably be added.

asl.library

It would probably be a good idea to add a print requester to asl.library. Many programs print (i.e. writing, drawing, drafting, or terminal programs). Having a print requester inside of asl would allow new features (such as multiple printers and improved graphic support) to be added seamlessly.

The application would use an addition gadget on the asl.library requester to select which printer to print to (like the PrtMgr's printer gadget). The application would be able to select one of the established printers, and modify certain text/graphic options.

Of course, only preferences can modify the printer driver and printer port.

The Printer prefs program uses a built-in requester that would be replaced by the asl.library requester.

1.7 V42 Printer Subsystem - Printer Manager

The Printer Manager provides control of the printers and the print jobs.

Gadget Controls

Printer Selection - Provides a list of printers that have been created through preferences. The one shown in the cycle gadget is the "current printer" that relates to the other gadgets and menus (such as Default, Reset, Print, ...).

Default - The \$DefaultPrinter environmental variable was created to support all the older programs that only know of printer.device unit 0. This check box will toggle between the current printer being the default (checked) and

unit 0 (unchecked). For unit 0, if this checkbox is checked, it will become disabled (as the toggle would be between unit 0 and unit 0).

Print Job List - From left to right, this list shows the print job ID, the name, the status, and the time started. The highlighted print job can be modified by the gadgets listed below.

Up/Down - A print job that is waiting can be moved up or down in priority. The only requirement is the print job can't be moved above a job that is printing. These gadgets are disabled as appropriate.

Status - The status cycle gadget is used to stop and restart a print job. The labels depend on the print job status. For example, if the print job is waiting, the cycle labels will be "Waiting" and "Stopped". If printing, they will be "Printing" and "Stopped". Clicking on this gadget will always cycle between not stopped and stopped, and the text will always be the print job's true status. (It's easier used than explained...).

Abort - This will mark the current print job as aborted.

Menu Controls

Print - `datatypes.library` is used to print a file using the currently selected printer. This can also be accomplished by dropping a file into the AppWindow.

About - Blah, blah, blah...

Hide - This will hide the window until the hot key (i.e. Ctrl-Alt-P) is pressed or the program is terminated.

Quit - The terminator...

Disabled - If checked, this printer will not accept any writes (i.e. the device/file will never be opened). The `printer.device` will still open as required for compatibility (i.e. programs that open the device for dimensions).

Paused - If checked, has the same effect as setting all the requests for this printer to stopped.

Reset - Write "`<ESC>#1`" to the current printer.

Form Feed - Write "`<FF>`" to the current printer.

Abort All - The same as selecting each print job for the current printer and clicking "Abort".

1.8 V42 Printer Subsystem - Preferences

The preferences program now includes a method of changing drivers and output devices (utilizing the PUNT chunk), and combines the functionality of the V3x text, graphics, and postscript preferences.

The exec device system is hidden from the user by using drop in icons (like the printers drivers) that contain device names. These are found

in the "DEVS:printerports" directory.

For example...

```
Icon name "Parallel 0"
DEVICE=parallel.device
UNIT=0                parallel.device unit 0
FLAGS=0              parallel.device flags
TYPE=PARALLEL       PP_PARALLEL=0
```

```
Icon name "Serial Solution 1"
DEVICE=ckptss.device
UNIT=1
FLAGS=0
TYPE=SERIAL         PP_SERIAL=1
```

```
Icon name "File"
TYPE=FILE          PP_FILE=3
```

```
Icon name "No Port"
TYPE=NULL         PP_NULL=4
```

The preferences program shows a list that contains the icon names. This is so users won't enter "scsi.device" as the output device.

NOTE:

The printergfx.prefs and printerps.prefs files have been merged into the printer.prefs file (printer.prefs now contains PUNT, PTXT, PGFX, PSPD chunks). To load an old printergfx.prefs file, run Printer, select the printer to modify, and load the printergfx.prefs file (either through "Open..." or by dragging the icon). This method also works with printerps.prefs.

Starting with Printer V42.22, it is also possible to do...

```
"Printer printergfx.prefs unit=0 save"
```

and with Printer V42.43...

```
"Printer printerps.prefs unit=0 save"
```

1.9 V42 Printer Subsystem - PRT: Handler

The V42 PRT: device allows for multiple units and support for most of the dos packets.

Each printer is considered a directory. Listing PRT: displays the unit number as the directory and the unit name in the comment field. Listing each directory will display the print job number as the file and the print job name for the comment.

A major difference between PRT: and normal file systems is the ability to write to a directory. The reason for this is the printer.device is the one to actually assign the print job number. For example, "DIR >PRT:1" will write to printer unit 1. The actual resulting file may be "PRT:1/3057". When choosing an output file name, the file part is ignored (i.e. "DIR >PRT:1/1" may still result with the file "PRT:1/3057"). Also, writing to "PRT:" is the same as writing to the \$DefaultPrinter.

Deleting a file will abort a print job. Directories, of course, can't be deleted.

The PRT: device does support file locks. As a result, it is possible to do "ASSIGN LASER: PRT:0". This would create the psuedo device "LASER:" which is actually printer.device unit 0. In addition, "CD PRT:" is allowed, along with changing directory to any of the subdirectories.

1.10 V42 Printer Subsystem - Includes and Autodocs

The "patches" directory contains patches for the three printer.device include files, and the printer autodocs. These patches are applied using the included "spatch" utility - which is Copyright © 1992 SAS Institute, Inc.

These patches should be applied to the V3.x includes found on the SAS/C V6.50 disk, along with the latest autodocs available from <mumble>.

IMPORTANT: Apply these patches to a backup copy. Future includes and autodocs will require the original C= files.