Apdf

Emmanuel Lesueur

Copyright © CopyrightÂ©1999 Emmanuel Lesueur

COLLABORATORS			
	TITLE : Apdf		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Emmanuel Lesueur	August 24, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Apdf

1.1 Apdf.guide

Apdf 1.3

Adapted from Xpdf 0.80 - ©1996-1998 Derek B. Noonburg

by Emmanuel Lesueur

1.2 Introduction

Apdf is a port of Derek B. Noonburg's PDF viewer, xpdf 0.80. It can read PDF version 1.2 files. This Amiga port has no support for Japanese fonts, nor for 16 bits characters. Due to legal issues, this release does not support encrypted documents. You can recompile Apdf to support it though (see source).

1.3 System requirements

To use Apdf, you need at least:

- AmigaOS 3.0

- a 68020 CPU

- MUI 3.8

- gzip 1.2.4 or something equivalent to the unix 'uncompress' command.

- for the PowerUp version: ppc.library 46

Recommended, but not required:

- util/misc/Type1_5.lha from Aminet, with

- some postscript fonts (see Fonts)

- a graphic card

- a fast CPU.

1.4 Installation

Simply put the Apdf executable wherever you want.
If you use the PowerUP version, also copy the Apdf.elf file in the same
directory.
Make sure that the "gzip" command is in your path, or see Configuration .
Note: for the PPC version you will probably need to either have gzip in
C: or specify its path explicitely.

1.5 Configuration

There are some parameters that can be adjusted. This is done by setting
tooltypes in Apdf's icon, or in the pdf documents icons.
The intended use is that one sets the "GZIPCMD", "URLCMD", "COLORS",
"DEFICON", "FONTMAP" (for default fonts), "DISKOBJECT", "CACHESIZE", and
"CACHEBLOCSIZE" tooltypes in Apdf's icon, and that other "FONTMAP"s, and
possibly "ZOOM" are set in document icons. Having document specific
cache parameters can make sense, too.
For an explanation of supported tooltypes, see here .

1.6 Starting Apdf

From CLI:
Just type "Apdf" in a shell to start Apdf. A requester asking for a
document will open. To view the file "myfile.pdf", you can also type

"Apdf myfile.pdf".
The complete command template is:
PDFFILE,P=PAGE/N/K,Z=ZOOM/N/K,C=COLORS/N/K,G=GZIPCMD/K,
U=URLCMD/K,I=DEFICON/K,F=FONTMAP/M,D=DISKOBJECT/K,
S=CACHESIZE/N/K,B=CACHEBLOCSIZE/N/K
PDFFILE is the name of the file to display.
PAGE is the page to display. Defaults to 1.
ZOOM is the zoom factor, from -5 to 5. Defaults to 1.
COLORS tells the maximum number of colors that Apdf should
allocate to show an embedded picture. Defaults to 16.
This parameter is ignored on HiColor/TrueColor screens.
GZIPCMD is the command used to uncompress data. Defaults to
"gzip -d -q".
URLCMD is the command used for external links. It should
contain a '%s', that will be replaced by the URL.
Defaults to "OpenURL %s".
DEFICON is the name of the default icon for pdf files. Apdf
will add this icon to a document that has none if you
try to save a document specific font mapping for it.
Defaults to "env:sys/def_pdf".
FONTMAP gives informations on which amiga fonts to substitute
for pdf fonts. Usually, you don't need to use that
option (see Configuration).
The syntax for this argument is detailed here .
DISKOBJECT is the name of the icon used when Apdf is iconified.
Defaults to Apdf's icon. The tooltypes of this icon
are not parsed.
CACHESIZE is the maximum size (in KB) of the memory used as cache.
0 means load the whole document in memory. Defaults to
256.
CACHEBLOCSIZE is the minimum amount of data (in KB) loaded during a
disk access. Ignored if CACHESIZE is 0. Defaults to 4.
Before parsing the arguments, Apdf will look in its icon and then in the
document's icon for tooltypes with those names (except "PDFFILE"). If
some are found, their values override the defaults values.
From Workbench:
To start Apdf from the Workbench, double click on its icon. This will
open a requester where you can select the document do view. You can also
click once on Apdf's icon and double click on the PDF document while

holding the shift key, or Apdf can be used as the default tool of a PDF
document.
Apdf supports the "PAGE", "ZOOM", "COLORS", "GZIPCMD", "URLCMD",
"DEFICON", "FONTMAP", "DISKOBJECT", "CACHESIZE", and "CACHEBLOCSIZE"
tool types with the same meaning as the corresponding cli arguments
(except that there should be one "FONTMAP" line per font mapping). The
tool types stored in Apdf's icon are read first, and are overridden by
those of the document's icon.
To quit Apdf, just close its window. You can also use the commodity
interface, send a "QUIT" command from ARexx, or send a CTRL-C signal.

1.7 FONTMAP

FONTMAP arguments use the following format:
pdf_font_name/[flags]amiga_font_name
Possible flags are:
'B' for a bold style.
'I' for an italic style.
'1' if the Amiga font uses a 'latin-1' encoding (default).
'2' if the Amiga font uses a 'latin-2' encoding.
's' if the Amiga font uses a 'symbols' encoding.
'z' if the Amiga font uses a 'dingbats' encoding.
'%x' to scale the amiga font of a factor x%, where x is between
1 and 999 (defaults=100).
Do not put a '1' or '2' flag just after a '%' flag.

1.8 Usage

1.9 Fonts

Apdf uses the standard Amiga font system to display text. To do that, it maps postscript fonts used by the document to amiga fonts, scaling them as appropriate.
By default, it uses the following fonts: times, helvetica, and courier.
But you can substitute other fonts if you like (see here).
Since those are bitmap fonts, scaling them usually does not give very nice results. To make the display nicer and more readable, you should use vector fonts instead, like Compugraphic, or Postscript ones.
AmigaOS can directly handle Compugraphic fonts, via the bullet.library. The CGTimes and CGTriumvirate fonts that come with the system are example of those. See the documentation of Intellifont in AmigaOS manuals for information on how to add other ones.
The Amiga system does not contain the necessary software to handle Postscript fonts. In order to use those, you need to install a Postscript font engine, like the one in the Type1_5.lha archive on Aminet. With this system, you can use any Postscript Type 1 font. You can find postscript fonts in the pix/pfont directory of Aminet, or in the ghostscript archive, or from some wordprocessors.
There is also a font engine for True Type fonts on Aminet.
AmigaOS has no equivalent of the "Symbols" and "Dingbats" postscript fonts. So unless you install some, you will miss some characters in pdf documents that use those. A Type1 version of theese fonts comes with Ghostscript.

1.10 Cache

Apdf implements a basic caching system to speed up document loading.
This system can be controlled by two parameters, CACHESIZE and CACHEBLOCSIZE, that represent memory amounts in Kbytes. Setting them to "good" values can improve Apdf's performance a lot.
CACHESIZE determines the maximum amount of memory that is used by Apdf for caching purposes. If CACHESIZE is zero, or if a document's size is smaller than CACHESIZE, the document will be fully loaded in memory and kept there until you quit Apdf or load another document.
If CACHESIZE if non-zero or if the document is too big, Apdf will keep at most N=CACHESIZE/CACHEBLOCSIZE blocs of BLOCSIZE Kbytes in memory.
When a part of the file that is not in memory is needed and Apdf already

has the maximum number of blocs, it frees the one that was not accessed
for the longest period, and replaces it by the new one.
In any cases, Apdf won't use more memory than what is actually needed.
The parameters that give the best performance depend on a lot of
factors, such as:
- the disk speed,
- the filesystem (FFS is slooow)
- the parameters of the filesystem (bloc size, buffers,...),
- whether or a caching program is used,
- the speed of the processor,
- the particular document you are trying to load.
Depending on your system, you'll have to try different settings to find
which parameters are the best for you. Some rules of thumb are:
- If you have lots of memory and don't have a too slow disk, setting
CACHESIZE=0 is probably the fastest.
- otherwise, the bigger CACHESIZE is the better.
- using too big values for CACHEBLOCSIZE can reduce the speed.
- the ratio CACHESIZE/CACHEBLOCSIZE should be "big enough".
By default, Apdf uses CACHESIZE=256 and CACHEBLOCSIZE=4.

1.11 Main window

Nothing complex here.
The current page of the document is displayed in the upper part of the
window. Dragging the mouse with the left button pressed selects a part
of the text. You can copy the text selected to the clipboard with the
'Copy' menu.
If the mouse is moved over a link, its destination will be displayed in
the lower right corner of the window. Clicking on the link activates it.
If the link is an URL, the command specified with the URLCMD option is
called.
The gadgets let you choose the current page and zoom factor.
You can enter a string to search in the 'search' gadget. The search is
not case sensitive. It starts after the current selected area, or at the
top of the page if nothing is selected. If the end of the document is
reached before the string is found, the begining of the document will be
searched, until the position where the search started. If the string is
found, it will be selected. Pressing the 'next' button will search the
next occurence of the same string.
Dropping an icon in the main window loads the corresponding document.

1.12 Menus

Project:
"About" - show some informations about Apdf/Xpdf.
"About MUI" - show the standard MUI information window.
"Open" - open a requester asking for a new PDF file to load.
"Save as" - open the Save window .
"Print" - open the Save window with filename
set to "PRT:" and mode set to "Postscript".
Attn: you must have a Postscript printer, or a
device that transparently converts postscript for
your printer for this to work.
"Quit" - quit.
Edit:
"Copy" - copy the selected text in the clipboard.
Settings:
"Default font mapping" - open the global fontmap editor .
"Document font mapping" - open the document specific fontmap editor .
"MUI" - open the MUI preference editor.

1.13 Save window

This window allows you to select parameters to save a part of a PDF
document in another format.
There are several possible save modes:
In "text" mode, all the graphics/pictures/fonts... are ignored. The
resulting file will contain plain ASCII text.
In "image" mode, the text is ignored. In this mode, apdf outputs one
file per image in the selected pages. If the filename selected in the
"file" gadget is "abc", the resulting files will be named "abc-nnn.xxx",
where nnn is a number and xxx is either "ppm", "pbm", or "jpg",
according to the picture format used. If the selected mode is
"PBM/PPM/JPEG", and a picture is stored in the DCT format in the PDF
file, it is saved as a jpeg file. Otherwise, monochrome pictures are
saved as PBM files and non-monochrome ones as PPM files. There are
datatypes on Aminet for all those formats.
In "Postscript" mode, both text and graphics are saved. The "Postscript
level 1" mode creates files that are bigger and in which images are
converted to black and white, but which can be used with level 1
printers.
The zoom and rotation parameters are only used in Postscript mode.

1.14 Font map windows

The font map windows allow you to customize the mapping between PDF fonts and amiga fonts.

The "default font mapping" window is used to customize global preferences, that will be saved in Apdf's icon. The "document font mapping" window is used for document specific preferences. Its contents is saved in the document's icon.

Each window contains a listview and several gadgets to edit the active entry in the listview. The first column contains the name of a PDF font, and the others describe the name of the Amiga font that should be substituted, a scaling factor (in %), two flags (bold and italic) that tell which algorithmic transformation to apply to the font, and the encoding used by the font (usually ISO-latin-1). The entries can be moved between the listviews by drag and drop.

For example, by default, the "Times-Bold" PDF font is mapped to the "times.font" Amiga font, with a scaling factor of 100%, bold style, and latin-1 encoding. If you wish to use the "CGTimes.font" Amiga font instead, select the "Times-Bold" line by clicking on it, and change "times.font" into "CGTimes.font" in the second string gadget. Since the "CGTimes.font" looks a bit smaller than the "times.font", you could give it a scaling factor of e.g. 115%.

There are 12 default fonts that are substituted for PDF fonts that don't have a matching entry in either table. Those fonts have names enclosed in « ».

For each PDF font used in a document, Apdf looks in the "document font map" for a corresponding entry. If one is found, the given Amiga font is used. Otherwise, it looks in the "default font map". If no entry is found here, Apdf examines the caracteristics of the font: serif/ sans-serif/fixed width/bold/italic, and looks again in both tables for the default font with the same caracteristics.

At the bottom of both windows, there are three buttons: Save, Apply, and scan.

The "Save" button saves the contents of the listview in either Apdf's icon or the document's icon, depending on the window. This is done by replacing all the FONTMAP tooltypes by others describing the contents of the listview. Other tooltypes are left unchanged.

The "Apply" button redraws the current page according to the current contents of both fontmap windows. The window in which "Apply" is pushed

makes no difference, the contents of both is considered.
The "Scan" button acts differently in each window. In the "default font map window", pushing scan adds to the listview the current mapping of 14 standard PDF fonts (Helvetica, Helvetica-Bold, ...) and the 12 default PDF fonts (« Sans-serif », ...). In the "document font map window", pushing "Scan" opens a window that lets you select a first page and a last page. Then the selected page range of the loaded document is searched, and for each PDF font used in those pages an entry is added to the listview, with the current mapping of this font.
Some things to consider to get good looking documents:
- Replace fonts with ones with similar properties (serif/sans-serif/ fixed width/italic/bold).
- If a font looks too big or too small, adjust the scaling factor.
- Vector fonts look better than bitmap fonts, but are a bit slower to load/scale.
- Do not set the bold/italic flag for fonts that are designed as bold or italic. E.g., if you have a "Helvetica-bold.font", it already is bold, so you don’t need to apply the algorithmic "bold" transformation.
- Fonts that are designed as bold/italic look better than those that are algorithmically transformed.

1.15 Keyboard shortcuts

The following shortcuts are supported in the main window when the active object is the document page. That means that if a string gadget is active you have to deactivate it first with the appropriate MUI shortcut (control-tab by default).
space - scroll down, skipping to next page if needed.
backspace - scroll up, returning to previous page if needed.
’o’ (open) - same as the "Open" menu item.
’f’ (find) - activate the ’search’ gadget.
’q’ (quit) - quit.
’+’ - increase the zoom factor.
’-’ - decrease the zoom factor.
The following MUI shortcuts (configured by the MUI preference editor) are also recognized:
Up - scroll up, returning to previous page if needed.
Down - scroll down, skipping to next page if needed.
PgUp - previous page.
PgDn - next page.
Home - first page.
End - last page.

1.16 MUI

Apdf using MUI, it supports all features common to MUI applications,
such as configuration, screen handling, iconification, commodity
interface, arexx... See MUI documentation for details.

1.17 Questions and answers

Q: Could you make a version supporting encrypted documents ?
A: No. The French law on cryptography software is currently as
restrictive as the American one. If you want to build such a version,
get the source archive, apply the xpdf-decryption.patch as explained
there, and recompile. If someone in a country with no restriction on
cryptography wants to build and make available an encryption-aware Apdf,
this is fine with me.
Q: Apdf only shows blank pages. What's the problem ?
A: It probably can not find gzip. Make sure it is in your path, or
specify its exact path in Apdf's icon (GZIPCMD tooltype). The PPC
version seems to have trouble finding it even when it really is in the
path. It should always find it in C:, though.

1.18 Legal informations

The PDF data structures, operators, and specification are copyright 1995
Adobe Systems Inc.
The original Xpdf is © 1996-1998 by Derek B. Nooburg.
Amiga specific parts of Apdf are © 1999 by Emmanuel Lesueur.
Apdf uses MUI. MUI is © 1992-1997 by Stefan Stuntz.
Due to the Unisys LZW patent, compressed data is handled by the extern
program gzip.
Due to various legislation on cryptography software, this version of
Apdf does not support encrypted documents.

1.19 Distribution

Apdf is released under the GNU General Public License version 2.

1.20 Source

The source code for Apdf can be found on Aminet, in the
gfx/show/Apdfsrc.lha archive.
** Warning **
Compiling Apdf is not an easy task. It requires a good knowledge of the
geekgadget environment. In particular, setting up an environment
allowing to compile the PPC version can be tricky.
With this archive and the decryption patch for xpdf-0.80 pointed to on
the page http://www.foolabs.com/xpdf/, you can build a version of Apdf
supporting encrypted documents.
With this archive and the gzip-1.2.4 source code, you can build a
version of Apdf with decompression handled internally. This version is a
bit faster, especially on PPC (but don't hope too much from it).
Compiling a 68k version of Apdf requires a GeekGadgets environnement
including:
- egcs C and C++ compilers (at least version 1.1)
- libnix
- amigaos includes
- make
- sh
- fileutils
and possibly a few others.
You also need the MUI and Cybergrapics developer kits.
To compile a PPC version of Apdf, you need all of the above, and
- egcs C and C++ compilers (at least version 1.1) for PPC
- libnixppc
- the powerup developer kit.
See the source archive for details.

1.21 Support

Send bug reports to lesueur@club-internet.fr.
Please make sure that you have read the Questions and answers section, first.
The main WWW address for the original xpdf is
http://www.foolabs.com/xpdf/

1.22 Future

While I will try to fix bugs, I will probably not add many enhancements
to Apdf.
Some ideas that would be nice to implement are:
- localization.
- WarpUp support.
- a PDF datatype.
I may localize the GUI, but won't localize the error messages of the
decoder. I have currently no plans to do the two other items.

1.23 History

1.3
- Fixed bug: Apdf used to scan one page more than it should in
the fontmap/scan function. Could lead to crash/enforcer hits
if the last page of the document was included in the scan range.
- Fixed bug: Apdf put the internal defaults in the default fontmap,
overriding the user defaults.
- Now calls GetDiskObjectNew to get the default icon for a document
that has none, when the icon specified by the DEFICON parameter
does not exist.
- Added a menu item shortcut for iconification.
- Fixed a bug in the gzip patch.
- Increased the update rate of the progress gauges.
- Added CPU informations in the "About" requester.
- Now checks that it is run on the correct CPU (may still crash
on 68000 or 68010, though).
- Worked around a bug in the Workbench "default tool" code.
- Fixed the DEFICON default value.
1.2 (some parts of the preceding release were incorectly marked 1.1,
so this release is 1.2 to avoid confusion)
- Added a 68020/68030 archive, since the 68060 doesn't work
on those processors.
- Added a basic cache system. Improves the speed a lot !
- Starting Apdf with no argument now opens a requester.
- Added support for documents with variable page size.
- Added support for rotated pages.
- Added progress bar and abort to the "search", "save", and "scan"

functions.

- URLCMD now defaults to "OpenURL %s".

- Added Up/Down MUI shortcuts.

- Added AppWindow support.

- Added the DISKOBJECT argument.

- Page changes scroll back to the top of the page.

- Fontmap scans no longer add several times the same entry.

- The scan window now opens with sliders preset to the current page.

- Added a "Print" menu, equivalent to "Save as" PRT: in postscript
mode. Attn: this will only work if you have a Postscript printer
or a printer driver that can interpret Postscript.

- The PPC task is no longer started at priority -1.

1.0-1.1

- First release.