Package: pdftools (via r-universe)

June 1, 2026

Type Package

Title Text Extraction, Rendering and Converting of PDF Documents

Version 3.9.0

Description Utilities based on 'libpoppler' <https://poppler.freedesktop.org> for extracting text, fonts, attachments and metadata from a PDF file. Also supports high quality rendering of PDF documents into PNG, JPEG, TIFF format, or into raw bitmap vectors for further processing in R.

License MIT + file LICENSE

URL https://ropensci.r-universe.dev/pdftools, https://docs.ropensci.org/pdftools/

BugReports https://github.com/ropensci/pdftools/issues

SystemRequirements Poppler C++ API: libpoppler-cpp-dev (deb) or poppler-cpp-devel (rpm), and poppler-data (rpm/deb) package.

Encoding UTF-8

Imports Rcpp (>= 0.12.12), qpdf

LinkingTo Rcpp

Suggests png, webp, tesseract, testthat

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Config/pak/sysreqs libjpeg-dev libssl-dev libpoppler-cpp-dev poppler-data

Repository https://rorylawless.r-universe.dev

Date/Publication 2026-05-13 20:38:22 UTC

RemoteUrl https://github.com/ropensci/pdftools

RemoteRef HEAD

RemoteSha 75f7912675b9b28e1bc1dc8f8e9d7d163450700e

Contents

`pdf_ocr_text` *OCR text extraction*

Description

Perform OCR text extraction. This requires you have the `tesseract` package.

Usage

```
pdf_ocr_text(
  pdf,
  pages = NULL,
  opw = "",
  upw = "",
  dpi = 600,
  language = "eng",
  options = NULL
)

pdf_ocr_data(
  pdf,
  pages = NULL,
  opw = "",
  upw = "",
  dpi = 600,
  language = "eng",
  options = NULL
)
```

Arguments

`pdf`	file path or raw vector with pdf data
`pages`	which pages of the pdf file to extract
`opw`	string with owner password to open pdf
`upw`	string with user password to open pdf
`dpi`	resolution to render image that is passed to pdf_convert.
`language`	passed to tesseract to specify the languge of the engine.
`options`	passed to tesseract to specify OCR parameters

See Also

Other pdftools: pdftools, qpdf, rendering

---

pdftools *PDF utilities*

Description

Utilities based on libpoppler for extracting text, fonts, attachments and metadata from a pdf file.

Usage

```
pdf_info(pdf, opw = "", upw = "")

pdf_text(pdf, opw = "", upw = "", raw = FALSE)

pdf_data(pdf, font_info = FALSE, opw = "", upw = "")

pdf_fonts(pdf, opw = "", upw = "")

pdf_attachments(pdf, opw = "", upw = "")

pdf_toc(pdf, opw = "", upw = "")

pdf_pagesize(pdf, opw = "", upw = "")
```

Arguments

pdf	file path or raw vector with pdf data
opw	string with owner password to open pdf
upw	string with user password to open pdf
raw	return text in raw stream order. Default is to use physical layout order.
font_info	if TRUE, extract font-data for each box. Be careful, this requires a very recent version of poppler and will error otherwise.

Details

The pdf_text function renders all textboxes on a text canvas and returns a character vector of equal length to the number of pages in the PDF file. On the other hand, pdf_data is more low level and returns one data frame per page, containing one row for each textbox in the PDF.

Note that pdf_data requires a recent version of libpoppler which might not be available on all Linux systems. When using pdf_data in R packages, condition use on `poppler_config()$has_pdf_data` which shows if this function can be used on the current system. For Ubuntu 16.04 (Xenial) and 18.04 (Bionic) you can use the PPA with backports of Poppler 0.74.0.

Poppler is pretty verbose when encountering minor errors in PDF files, in especially pdf_text. These messages are usually safe to ignore, use suppressMessages to hide them altogether.

See Also

Other pdftools: pdf_ocr_text(), qpdf, rendering

Examples

```
# Just a random pdf file
pdf_file <- file.path(R.home("doc"), "NEWS.pdf")
info <- pdf_info(pdf_file)
text <- pdf_text(pdf_file)
fonts <- pdf_fonts(pdf_file)
files <- pdf_attachments(pdf_file)
```

---

rendering *Render / Convert PDF*

Description

High quality conversion of pdf page(s) to png, jpeg or tiff format, or render into a raw bitmap array for further processing in R.

Usage

```
pdf_render_page(
  pdf,
  page = 1,
  dpi = 72,
  numeric = FALSE,
  antialias = TRUE,
  opw = "",
  upw = ""
)

pdf_convert(
  pdf,
  format = "png",
  pages = NULL,
  filenames = NULL,
  dpi = 72,
  antialias = TRUE,
  opw = "",
  upw = "",
  verbose = TRUE
)

poppler_config()
```

Arguments

pdf	file path or raw vector with pdf data
page	which page to render
dpi	resolution (dots per inch) to render
numeric	convert raw output to (0-1) real values
antialias	enable antialiasing. Must be "text" or "draw" or TRUE (both) or FALSE (neither).
opw	owner password
upw	user password
format	string with output format such as "png" or "jpeg". Must be equal to one of poppler_config()$supported_image_formats.
pages	vector with one-based page numbers to render. NULL means all pages.
filenames	vector of equal length to pages with output filenames. May also be a format string which is expanded using pages and format respectively.
verbose	print some progress info to stdout

See Also

Other pdftools: pdf_ocr_text(), pdftools, qpdf

Examples

```
# Rendering should be supported on all platforms now
# convert few pages to png
file.copy(file.path(Sys.getenv("R_DOC_DIR"), "NEWS.pdf"), "news.pdf")
pdf_convert("news.pdf", pages = 1:3)

# render into raw bitmap
bitmap <- pdf_render_page("news.pdf")

# save to bitmap formats
png::writePNG(bitmap, "page.png")
webp::write_webp(bitmap, "page.webp")

# Higher quality
bitmap <- pdf_render_page("news.pdf", page = 1, dpi = 300)
png::writePNG(bitmap, "page.png")

# slightly more efficient
bitmap_raw <- pdf_render_page("news.pdf", numeric = FALSE)
webp::write_webp(bitmap_raw, "page.webp")

# Cleanup
unlink(c('news.pdf', 'news_1.png', 'news_2.png', 'news_3.png',
 'page.jpeg', 'page.png', 'page.webp'))
```

Index