

# Package: doclingr (via r-universe)

July 10, 2026

**Title** Document Intelligence via 'Docling'

**Version** 0.1.0

**Description** An interface to 'Docling', a document-understanding library that converts 'PDF', 'DOCX', 'PPTX', 'HTML' and image documents into structured, AI-ready data. The package wraps the 'Docling' 'Python' package through 'reticulate' to extract layout-aware text, tables and metadata, export to 'Markdown' or 'JSON', and split documents into context-rich chunks suitable for retrieval-augmented generation (RAG) and embedding pipelines.

**License** MIT + file LICENSE

**Language** en-US

**Encoding** UTF-8

**RoxygenNote** 8.0.0

**URL** <https://github.com/StrategicProjects/doclingr>,  
<https://strategicprojects.github.io/doclingr/>

**BugReports** <https://github.com/StrategicProjects/doclingr/issues>

**SystemRequirements** Python (>= 3.9), docling (>= 2.20.0; tested with 2.107.0)

**Imports** reticulate (>= 1.34.0), cli, rlang, stats, tibble

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Andre Leite [aut, cre], Marcos Wasilew [aut], Hugo Vasconcelos [aut], Carlos Amorim [aut], Diogo Bezerra [aut]

**Maintainer** Andre Leite <leite@castlab.org>

**Config/pak/sysreqs** python3

**Repository** <https://cran.r-universe.dev>

**Date/Publication** 2026-07-10 20:30:26 UTC

**RemoteUrl** <https://github.com/cran/doclingr>

**RemoteRef** HEAD

**RemoteSha** 827d89d1050de616cfab47ea1c80e3b7819eeea8

## Contents

docling_available . . . . .	2
docling_chunk . . . . .	3
docling_convert . . . . .	4
docling_embed . . . . .	6
docling_export . . . . .	7
docling_figures . . . . .	8
docling_n_pages . . . . .	9
docling_tables . . . . .	9
install_docling . . . . .	10
<b>Index</b>	<b>12</b>

---

docling_available	<i>Is the Docling backend available?</i>
-------------------	--

---

### Description

Checks whether the docling Python package can be imported in the active 'reticulate' environment.

### Usage

```
docling_available()
```

### Value

A logical scalar.

### See Also

[install\\_docling\(\)](#)

### Examples

```
## Not run:
docling_available()

## End(Not run)
```

docling\_chunk

*Split a document into RAG-ready chunks***Description**

Apply a Docling chunker to a converted document and return the chunks as a tidy tibble. The default "hybrid" chunker produces tokenization-aware, context-enriched chunks well suited to embedding and retrieval pipelines; the "hierarchical" chunker follows the document's structural hierarchy without a token budget.

**Usage**

```
docling_chunk(
  x,
  chunker = c("hybrid", "hierarchical"),
  tokenizer = NULL,
  max_tokens = NULL,
  contextualize = TRUE,
  ...
)
```

**Arguments**

x	A docling_document from <code>docling_convert()</code> .
chunker	Either "hybrid" (default) or "hierarchical".
tokenizer	Hugging Face model id whose tokenizer is used to count tokens (hybrid chunker only). Defaults to a small sentence-embedding tokenizer when <code>max_tokens</code> is set; NULL uses Docling's built-in default.
max_tokens	Optional integer token budget per chunk (hybrid chunker only). When NULL, the tokenizer's own maximum is used.
contextualize	When TRUE (default), each chunk's text is enriched with surrounding headings and table context via the chunker's <code>contextualize()</code> method — the form you typically embed. The raw text is always also returned in <code>raw_text</code> .
...	Additional keyword arguments forwarded to the Python chunker constructor (for example <code>merge_peers</code> or <code>repeat_table_header</code> ).

**Details**

The hybrid chunker is token-aware: it packs content up to a token budget and splits oversized passages. Control this with `tokenizer` (the model whose tokenizer defines "a token") and `max_tokens` (the budget). These are ignored by the hierarchical chunker.

**Value**

A `tibble::tibble` with one row per chunk and columns:

- `chunk_id` — 1-based index.
- `text` — contextualized text (or raw text if `contextualize = FALSE`).
- `raw_text` — the chunk's unmodified text.
- `n_chars` — number of characters in `text`.
- `headings` — list-column of heading paths for the chunk.
- `pages` — list-column of integer page numbers the chunk spans.
- `n_doc_items` — number of underlying document items in the chunk.

**See Also**

`docling_convert()`, `docling_embed()`

**Examples**

```
## Not run:
doc <- docling_convert("paper.pdf")
chunks <- docling_chunk(doc, max_tokens = 512)
chunks$text[1]

# Match your embedding model's tokenizer
docling_chunk(doc, tokenizer = "BAAI/bge-small-en-v1.5", max_tokens = 512)

## End(Not run)
```

---

docling\_convert

*Convert one or more documents with Docling*

---

**Description**

Runs Docling's document-understanding pipeline over local file paths or URLs and returns lightweight R handles around the resulting DoclingDocuments. Each handle can be exported to Markdown or JSON (`as_markdown()`, `as_json()`), mined for tables (`docling_tables()`), or split into chunks for RAG (`docling_chunk()`).

**Usage**

```
docling_convert(
  source,
  ocr = TRUE,
  table_mode = c("accurate", "fast"),
  device = c("auto", "cpu", "cuda", "mps"),
  num_threads = NULL,
  images = FALSE,
```

```

    images_scale = 1,
    ...
)

```

### Arguments

source	A character vector of file paths and/or URLs. A single source returns one <code>docling_document</code> ; multiple sources are converted in one batch and returned as a <code>docling_document_list</code> .
ocr	Logical; run OCR on the document. Defaults to TRUE. Set to FALSE to skip OCR for faster conversion of born-digital documents.
table_mode	Table-structure model mode, one of "fast" or "accurate" (default). "accurate" produces better table structure at some speed cost.
device	Accelerator device for the deep-learning models: "auto" (default), "cpu", "cuda", or "mps".
num_threads	Optional integer number of CPU threads for the accelerator. NULL (default) leaves Docling's default.
images	Logical; generate and retain page and picture images so they can be saved later with <code>docling_figures()</code> . Defaults to FALSE (smaller, faster results). Required if you want image files out.
images_scale	Image resolution scale relative to 72 DPI when <code>images</code> is TRUE (for example 2 is approx. 144 DPI). Defaults to 1.
...	Reserved for future pipeline options; currently ignored with a warning if supplied.

### Details

Supported inputs include PDF, DOCX, PPTX, XLSX, HTML, Markdown, AsciiDoc and common image formats, as determined by the installed Docling version.

### Value

For a single source, an object of class `docling_document` (a list with the underlying Python document and the original source). For multiple sources, a `docling_document_list`: a list of `docling_document` objects named by source.

### See Also

[as\\_markdown\(\)](#), [docling\\_tables\(\)](#), [docling\\_chunk\(\)](#)

### Examples

```

## Not run:
doc <- docling_convert("https://arxiv.org/pdf/2408.09869")
as_markdown(doc)

# Batch, OCR off, fast tables
docs <- docling_convert(c("a.pdf", "b.pdf"), ocr = FALSE, table_mode = "fast")

```

```
## End(Not run)
```

---

```
docling_embed      Attach embeddings to chunks
```

---

### Description

Embed a chunk tibble's text with a user-supplied embedding function and return the tibble with an embedding list-column. `doclingr` stays provider-agnostic: you bring the embedder (an OpenAI/Cohere/Ollama API call, a local sentence-transformers model via `reticulate`, anything), and this helper handles batching, validation and tidy assembly.

### Usage

```
docling_embed(chunks, embedder, text_column = "text", batch_size = NULL)
```

### Arguments

<code>chunks</code>	A tibble of chunks, typically from <code>docling_chunk()</code> .
<code>embedder</code>	A function taking a character vector and returning either a numeric matrix with one row per input, or a list of equal-length numeric vectors.
<code>text_column</code>	Name of the column to embed. Defaults to "text".
<code>batch_size</code>	Optional integer; if set, <code>embedder</code> is called on successive batches of at most this many texts and the results concatenated. Useful for APIs with per-request limits. NULL (default) embeds in one call.

### Value

`chunks` with an added embedding list-column of numeric vectors, and an `n_dim` integer column giving each embedding's length.

### See Also

[docling\\_chunk\(\)](#)

### Examples

```
## Not run:
chunks <- docling_chunk(docling_convert("paper.pdf"), max_tokens = 512)

# Any embedder: here a toy one
embed_fn <- function(txt) matrix(stats::runif(length(txt) * 8), nrow = length(txt))
docling_embed(chunks, embed_fn)

## End(Not run)
```

---

docling_export	<i>Export a converted document</i>
----------------	------------------------------------

---

## Description

Render a `docling_convert()` result into a downstream-friendly format.

## Usage

```
as_markdown(x, ...)  
  
## S3 method for class 'docling_document'  
as_markdown(x, ...)  
  
as_text(x, ...)  
  
## S3 method for class 'docling_document'  
as_text(x, ...)  
  
as_html(x, ...)  
  
## S3 method for class 'docling_document'  
as_html(x, ...)  
  
as_json(x, ...)  
  
## S3 method for class 'docling_document'  
as_json(x, ...)  
  
as_doctags(x, ...)  
  
## S3 method for class 'docling_document'  
as_doctags(x, ...)
```

## Arguments

x	A <code>docling_document</code> from <code>docling_convert()</code> .
...	Additional arguments passed to the underlying Docling export method (for example <code>image_mode</code> for Markdown).

## Value

- `as_markdown()`: a length-1 character string of Markdown.
- `as_text()`: a length-1 character string of plain text.
- `as_html()`: a length-1 character string of HTML.
- `as_json()`: an R list mirroring the `DoclingDocument` structure.
- `as_doctags()`: a length-1 character string in Docling's `DocTags` format.

**Examples**

```
## Not run:
doc <- docling_convert("report.pdf")
as_markdown(doc)
str(as_json(doc), max.level = 1)

## End(Not run)
```

---

docling\_figures

*Extract figures (pictures) from a converted document*


---

**Description**

Return a tidy tibble of the pictures Docling detected, with their captions and page numbers. When `image_dir` is supplied and the document was converted with `images = TRUE`, each picture is written to disk and its path returned.

**Usage**

```
docling_figures(x, image_dir = NULL, format = "png")
```

**Arguments**

<code>x</code>	A <code>docling_document</code> from <code>docling_convert()</code> .
<code>image_dir</code>	Optional directory to save picture images into. Created if it does not exist. Requires <code>docling_convert()</code> to have been called with <code>images = TRUE</code> ; otherwise image data is unavailable and <code>image_path</code> is <code>NA</code> with a warning.
<code>format</code>	Image file format when saving, for example "png" (default) or "jpeg".

**Value**

A `tibble::tibble` with one row per figure and columns:

- `figure_id` — 1-based index.
- `caption` — caption text (empty string if none).
- `page` — page number the figure appears on (NA if unknown).
- `image_path` — path to the saved image, or NA if not saved.

**See Also**

`docling_convert()`

**Examples**

```
## Not run:
doc <- docling_convert("paper.pdf", images = TRUE)
figs <- docling_figures(doc, image_dir = "figures")
figs$image_path

## End(Not run)
```

---

docling_n_pages	<i>Number of pages in a converted document</i>
-----------------	--

---

**Description**

Number of pages in a converted document

**Usage**

```
docling_n_pages(x)
```

**Arguments**

x                    A docling\_document from [docling\\_convert\(\)](#).

**Value**

An integer page count (0 for page-less formats such as Markdown).

**Examples**

```
## Not run:
docling_n_pages(docling_convert("report.pdf"))

## End(Not run)
```

---

docling_tables	<i>Extract tables as data frames</i>
----------------	--------------------------------------

---

**Description**

Pull every table detected by Docling out of a converted document and return them as a list of tibbles, preserving the document order.

**Usage**

```
docling_tables(x)
```

**Arguments**

x                    A docling\_document from `docling_convert()`.

**Value**

A list of `tibble::tibles`, one per detected table. The list is empty if no tables were found. Each element carries a page attribute when Docling reports the originating page.

**See Also**

`docling_convert()`

**Examples**

```
## Not run:
doc <- docling_convert("financials.pdf")
tbls <- docling_tables(doc)
tbls[[1]]

## End(Not run)
```

---

install\_docling

*Install the Docling Python backend*


---

**Description**

Installs the docling Python package (and its dependencies) into a 'reticulate'-managed environment. This is a thin wrapper around `reticulate::py_install()` that you typically run once after installing doclingr.

**Usage**

```
install_docling(
  envname = "r-docling",
  method = c("auto", "virtualenv", "conda"),
  extra = NULL,
  ...
)
```

**Arguments**

envname	Name of, or path to, the target Python environment. Defaults to "r-docling", created on first use.
method	Installation method passed to <code>reticulate::py_install()</code> : one of "auto", "virtualenv", or "conda".
extra	Optional character vector of additional pip/conda specs to install alongside Docling (for example "docling[ocr]" or a pinned version such as "docling==2.0.0").
...	Further arguments forwarded to <code>reticulate::py_install()</code> .

**Value**

Invisibly NULL, called for its side effect.

**See Also**

[docling\\_available\(\)](#), [docling\\_convert\(\)](#)

# Index

`as_doctags (docling_export)`, 7  
`as_html (docling_export)`, 7  
`as_json (docling_export)`, 7  
`as_json()`, 4  
`as_markdown (docling_export)`, 7  
`as_markdown()`, 4, 5  
`as_text (docling_export)`, 7

`docling_available`, 2  
`docling_available()`, 11  
`docling_chunk`, 3  
`docling_chunk()`, 4–6  
`docling_convert`, 4  
`docling_convert()`, 3, 4, 7–11  
`docling_embed`, 6  
`docling_embed()`, 4  
`docling_export`, 7  
`docling_figures`, 8  
`docling_figures()`, 5  
`docling_n_pages`, 9  
`docling_tables`, 9  
`docling_tables()`, 4, 5

`install_docling`, 10  
`install_docling()`, 2

`reticulate::py_install()`, 10

`tibble::tibble`, 4, 8, 10