Package: tynding (via r-universe)

October 4, 2026

Title 'Typst' Bindings

Version 0.6.0

Description Provides bindings to the 'Typst' typesetting system, enabling users to compile 'Typst' documents directly from R. The package interfaces with the 'Typst' 'Rust' library to render documents, making it possible to integrate 'Typst'-based workflows into R scripts, reports, and reproducible research pipelines.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://y-sunflower.github.io/tynding/

BugReports https://github.com/y-sunflower/tynding/issues

Depends R (>= 4.2)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/rextendr/version 0.5.0

SystemRequirements Cargo (Rust's package manager), rustc >= 1.89.0, xz

Imports jsonlite

Config/pak/sysreqs xz-utils libclang-dev

Repository https://y-sunflower.r-universe.dev

Date/Publication 2026-07-06 06:15:04 UTC

RemoteUrl https://github.com/y-sunflower/tynding

RemoteRef HEAD

RemoteSha de6fce1e8b0956867c419ba6cb3ef8eb53ce0fff

Contents

is_valid_typst 2
typst_compile 3
typst_write 4

Index **5**

`is_valid_typst` *Is valid Typst?*

Description

Check that a character vector is valid Typst markup by compiling it. If no error, it assumes the code is valid.

Usage

```
is_valid_typst(x, error_on_failure = FALSE)
```

Arguments

`x`	A character vector
`error_on_failure`	Whether to raise an error if the code is invalid. Default to `FALSE`.

Value

Indicates whether the output PDF file exists (for example, if `TRUE`, then Typst has been compiled successfully).

Examples

```
typst_code <- c("= Hello World", "This is a Typst document.")
is_valid_typst(typst_code) # TRUE

typst_code <- c("= Hello World", "#This is a Typst document.")
is_valid_typst(typst_code) # FALSE

## Not run:
typst_code <- c("= Hello World", "#This is a Typst document.")
is_valid_typst(typst_code, error_on_failure = TRUE) # ERROR

## End(Not run)
```

typst_compile *Compile a* `.typ` *file and return the output path.*

Description

This function uses the Typst Rust library to compile a `.typ` file to a supported output format and return the output path.

Usage

```
typst_compile(
  file,
  output = NULL,
  font_path = NULL,
  pdf_standard = NULL,
  output_format = NULL,
  root = NULL,
  ppi = NULL,
  ignore_system_fonts = FALSE,
  ...
)
```

Arguments

`file`	Path to an existing `.typ` file.
`output`	Optional output path. Defaults to the input path with the extension implied by the output format.
`font_path`	Optional path to font files.
`pdf_standard`	Optional PDF standard specification. Can be a single standard or a character vector of compatible standards. Options are: `1.4`, `1.5`, `1.6`, `1.7`, `2.0`, `a-1b`, `a-1a`, `a-2b`, `a-2u`, `a-2a`, `a-3b`, `a-3u`, `a-3a`, `a-4`, `a-4f`, `a-4e`, `ua-1`. Only used for PDF output.
`output_format`	Optional output format. Supported values are `pdf`, `html`, `png`, and `svg`. Defaults to `NULL`, which means "infer from `output` when possible, otherwise use `pdf`". For multi-page `png` and `svg` outputs, `output` must be a template path containing at least one of `{p}`, `{0p}`, or `{t}`.
`root`	Optional Typst project root. Defaults to the parent directory of `file`. When provided, `file` must be contained in that directory's subtree.
`ppi`	Optional pixels per inch value when exporting to png. If NULL, default to 144.0.
`ignore_system_fonts`	Whether to skip system font discovery. Embedded Typst fonts and fonts from `font_path` are still available.
`...`	Named inputs passed to the Typst document via `sys.inputs`. Each argument must be named. Scalar values are passed as-is; other values are JSON-encoded.

Value

Output path, invisibly.

`typst_write` *Write a Typst file from a character vector*

Description

Create a Typst file (.typ) from a character vector.

Usage

```
typst_write(x, output = NULL)
```

Arguments

`x`	A character vector representing Typst code.
`output`	Optional output file path (must end with ".typ"). If NULL, a temporary file is created.

Value

The path to the written .typ file, invisibly.

Examples

```
## Not run:
code <- c("= Hello World", "This is a Typst document.")
typst_write(code, output = "hello.typ")

## End(Not run)
```

Index

is_valid_typst, 2

typst_compile, 3
typst_write, 4