Package: blockr.io (via r-universe)

May 26, 2026

Title Interactive File Import and Export Blocks

Version 0.1.0.9000

Description Extends 'blockr.core' with interactive blocks for reading and writing data files. Supports CSV, Excel, Parquet, RDS, and other formats through a graphical interface without writing code directly. Includes file browser integration and configurable import/export options.

URL https://bristolmyerssquibb.github.io/blockr.io/

BugReports https://github.com/BristolMyersSquibb/blockr.io/issues

License GPL (>= 3)

Depends R (>= 4.1.0)

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Remotes BristolMyersSquibb/blockr.core

Imports blockr.core (>= 0.1.2), htmltools, jsonlite, readxl, shiny, readr, rio, shinyjs, writexl, zip

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

Config/testthat/edition 3

Config/testthat/parallel true

Config/pak/sysreqs cmake make libuv1-dev libssl-dev libx11-dev zlib1g-dev

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

Date/Publication 2026-05-11 17:26:05 UTC

RemoteUrl https://github.com/bristolmyerssquibb/blockr.io

RemoteRef HEAD

RemoteSha 78393761ab3465a68fadf87162a266623167d20d

Contents

`file_category` *File category from extension*

Description

Categorizes a file by its extension into a broad format family that determines reader dispatch and UI adaptation.

Usage

```
file_category(path)
```

Arguments

`path`	Character. File path.

Value

One of `"csv"`, `"excel"`, `"arrow"`, `"other"`.

`file_extensions` *Supported file extensions*

Description

Returns a character vector of file extensions (without dots) supported by the read block. Useful for sibling packages that need to filter or validate file paths before passing them to blockr.io.

Usage

```
file_extensions()
```

Value

Character vector of file extensions (without dots)

`new_data_dir_option` *Data directory board option*

Description

A board-level option that sets a default data directory for read and write blocks. When set, file paths entered in blocks are resolved relative to this directory.

Usage

```
new_data_dir_option(
  value = blockr_option("data_dir", ""),
  category = "Data",
  ...
)
```

Arguments

`value`	Character. Initial directory path. Default: empty string (no data directory).
`category`	Character. Option category for UI grouping.
`...`	Forwarded to `blockr.core::new_board_option()`.

Value

A `board_option` object.

`new_download_block` *Download-only file export block*

Description

A variadic block that lets the user download one or more data frames as a file in the browser. Intended as a lightweight counterpart to `new_write_block()` for the common case where no server-side save is needed.

Usage

```
new_download_block(filename = "", format = "csv", args = list(), ...)
```

Arguments

filename	Character. Optional fixed filename (without extension). If empty (default), a timestamped filename is generated.
format	Character. One of the values in write_formats(): "csv", "excel", "parquet", or "feather". Default: "csv".
args	Named list of format-specific writing parameters (same as new_write_block()). Only relevant values for the selected format are used.
...	Forwarded to blockr.core::new_transform_block().

Details

Multi-input behavior matches new_write_block(): multiple inputs produce a multi-sheet Excel file for format = "excel", or a ZIP archive for CSV, Parquet, and Feather.

Adding a new format (e.g. SAS) only requires extending write_formats(), format_extension(), and the dispatch in write_expr() - both new_write_block() and new_download_block() pick it up automatically.

Value

A blockr transform block exposing a download button.

Examples

```
if (interactive()) {
  library(blockr.core)
  serve(new_download_block())
}
```

---

new_read_block *Unified file reading block*

Description

A single block for reading files in various formats with smart UI that adapts based on detected file type. Supports "From Browser" (upload) and "Location" (path/URL input) modes with persistent storage for uploaded files.

Usage

```
new_read_block(
  path = character(),
  source = "upload",
  combine = "auto",
  args = list(),
  ...
)
```

Arguments

`path`	Character vector of file paths to pre-load. Accepts local paths and URLs. When provided, automatically switches to "path" mode regardless of the source parameter.
`source`	Either "upload" for file upload widget or "path" for path/URL input. Default: "upload". Automatically set based on path parameter.
`combine`	Strategy for combining multiple files: "auto", "rbind", "cbind", "first"
`args`	Named list of format-specific reading parameters. Only specify values that differ from defaults. Available parameters: • **For CSV files:** `sep` (default: ","), `quote` (default: '"'), `encoding` (default: "UTF-8"), `skip` (default: 0), `n_max` (default: Inf), `col_names` (default: TRUE) • **For Excel files:** `sheet` (default: NULL), `range` (default: NULL), `skip` (default: 0), `n_max` (default: Inf), `col_names` (default: TRUE)
`...`	Forwarded to `blockr.core::new_data_block()`

Details

File Handling Modes:

The block supports two modes:

From Browser mode (upload):

- User uploads files from their computer via the browser
- Files are copied to persistent storage directory (upload_path)
- State stores permanent file paths
- Works across R sessions with state restoration

Location mode (path):

- User enters a file path or URL in a text input with autocomplete
- For server paths: reads directly from original location
- For URLs: downloads to a temporary file each time
- When a board-level data directory is set, paths are resolved relative to it

Smart Adaptive UI:

After file selection, the UI detects file type and shows relevant options:

- **CSV/TSV:** Delimiter, quote character, encoding options
- **Excel:** Sheet selection, cell range
- **Other formats:** Minimal or no options (handled automatically)

Multi-file Support:

When multiple files are selected:

- **"auto"**: Attempts rbind, falls back to first file if incompatible
- **"rbind"**: Row-binds files (requires same columns)
- **"cbind"**: Column-binds files (requires same row count)
- **"first"**: Uses only the first file

Value

A blockr data block that reads file(s) and returns a data.frame.

Configuration

The following settings are retrieved from options and not stored in block state:

- **upload_path**: Directory for persistent file storage. Set via `options(blockr.upload_path = "/path")` or environment variable `BLOCKR_UPLOAD_PATH`. Default: `tools::R_user_dir("blockr", "data")`

Examples

```
# Create a read block for a CSV file
csv_file <- tempfile(fileext = ".csv")
write.csv(mtcars[1:5, ], csv_file, row.names = FALSE)
block <- new_read_block(path = csv_file)
block

# With custom CSV parameters
block <- new_read_block(
  path = csv_file,
  args = list(n_max = 3)
)

if (interactive()) {
  # Launch interactive app
  serve(new_read_block())
}
```

---

new_write_block *Unified file writing block*

Description

A variadic block for writing dataframes to files in various formats. Accepts multiple input dataframes and handles single files, multi-sheet Excel, or ZIP archives depending on format and number of inputs.

Usage

```
new_write_block(
  directory = "",
  filename = "",
  format = "csv",
  auto_write = FALSE,
  args = list(),
  mode = NULL,
```

```
  ...
)
```

Arguments

directory	Character. Default directory for file output. When non-empty, enables server-side writing. Can be configured via `options(blockr.write_dir = "/path")` or environment variable `BLOCKR_WRITE_DIR`. Default: `""` (empty — download-only until user sets a path).
filename	Character. Optional fixed filename (without extension). • **If provided**: Writes to same file path on every upstream change (auto-overwrite) • **If empty** (default): Generates timestamped filename (e.g., `data_20250127_143022.csv`)
format	Character. Output format: "csv", "excel", "parquet", or "feather". Default: "csv"
auto_write	Logical. When TRUE, automatically writes files when data changes (requires a non-empty directory). When FALSE (default), user must click "Save to File" button.
args	Named list of format-specific writing parameters. Only specify values that differ from defaults. Available parameters: • **For CSV files:** `sep` (default: ","), `quote` (default: TRUE), `na` (default: "") • **For Excel/Arrow:** Minimal options needed (handled by underlying packages)
mode	**[Deprecated]** Previously selected between "browse" and "download" tabs. Now ignored — both download and server-save are always available. Kept for backwards compatibility; emits a deprecation warning when non-NULL.
...	Forwarded to `blockr.core::new_transform_block()`

Details

Variadic Inputs:

This block accepts multiple dataframe inputs (1 or more) similar to `bind_rows_block`. Inputs can be numbered ("1", "2", "3") or named ("sales_data", "inventory"). Input names are used for sheet names (Excel) or filenames (multi-file ZIP).

File Output Behavior:

Single input:

- Writes single file in specified format
- Filename: `{filename}.{ext}` or `data_{timestamp}.{ext}`

Multiple inputs + Excel:

- Single Excel file with multiple sheets
- Sheet names derived from input names

Multiple inputs + CSV/Arrow:

- Single ZIP file containing individual files
- Each file named from input names

Filename Behavior:

Fixed filename (`filename = "output"`):

- Reproducible path: always writes to `{directory}/output.{ext}`
- Overwrites file on every upstream data change
- Ideal for automated pipelines

Auto-timestamped (`filename = ""`):

- Unique files: `{directory}/data_YYYYMMDD_HHMMSS.{ext}`
- Preserves history, prevents accidental overwrites
- Safe default behavior

Download vs Server Save:

Both options are always available in a flat layout (no tabs):

Download to Browser:

- Always available via the download button
- Triggers a download to your browser's download folder

Save to Server:

- Active when a server directory path is set (non-empty)
- User enters a directory path in the path input
- Files persist on server
- When running locally, this is your computer's file system

Value

A blockr transform block that writes dataframes to files

Examples

```
# Create a write block for CSV output
block <- new_write_block(
  directory = tempdir(),
  filename = "output",
  format = "csv"
)
block

# Write block for Excel with auto-timestamp
block <- new_write_block(
  directory = tempdir(),
  filename = "",
  format = "excel"
)

if (interactive()) {
  # Launch interactive app
  serve(new_write_block())
}
```

path_input *Path input widget*

Description

A Shiny module that provides a text input with server-side file/directory autocomplete. Used by the read and write blocks to replace shinyFiles browser widgets.

Usage

```
path_input_ui(id, prefix = NULL, upload_id = NULL)

path_input_server(
  id,
  data_dir = reactive(""),
  mode = c("file", "directory"),
  extensions = NULL
)
```

Arguments

id	Module namespace ID.
prefix	Optional initial prefix text shown before the input (typically the data directory path).
upload_id	Optional ID of a hidden Shiny `fileInput` to wire up for upload-icon click and drag-and-drop. When non-NULL, an upload icon button is rendered inside the input field and the container gets a `data-upload-target` attribute pointing to this ID.
data_dir	Reactive returning the current data directory path (from board options). Empty string means no data directory.
mode	Either `"file"` or `"directory"`. Controls which entries are selectable in the autocomplete dropdown.
extensions	Optional character vector of file extensions (without dots) to show in autocomplete. Defaults to `NULL`, which shows all rio-supported formats. Use e.g. `"rtf"` to restrict to RTF files only.

Value

`path_input_ui()` returns a `tagList` with the widget HTML. `path_input_server()` returns a `reactive` containing the current path text value.

read_file_expr *Create a read expression for a single file*

Description

Convenience wrapper that detects the file category from the path and returns an unevaluated R expression for reading the file. Useful for sibling packages that construct pipelines programmatically.

Usage

```
read_file_expr(path, ...)
```

Arguments

path	Character. Path to a single file.
...	Additional parameters forwarded to the reader (e.g. sep, sheet, skip).

Value

A language object (unevaluated call) that, when evaluated, reads the file into a data frame.

write_file_expr *Create a write expression for a single file*

Description

Convenience wrapper that detects the file format from the path extension and returns an unevaluated R expression for writing the data. Useful for sibling packages that construct pipelines programmatically (e.g. DM blocks).

Usage

```
write_file_expr(data, path, ...)
```

Arguments

data	Character. Name of the data object to write.
path	Character. Output file path (extension determines format).
...	Additional parameters forwarded to the writer (e.g. sep, quote, na for CSV files).

Value

A language object (unevaluated call) that, when evaluated, writes the data frame to the file.

Examples

```
write_file_expr("mtcars", "/tmp/cars.csv")
write_file_expr("iris", "/tmp/flowers.xlsx")
write_file_expr("df", "/tmp/data.parquet")
```

`write_formats` *Supported write formats*

Description

Returns a named character vector of supported output formats for writing data frames. Names are display labels, values are format identifiers. Used to populate format dropdowns in the write block UI.

Usage

```
write_formats()
```

Value

A named character vector.

Index