Package: excelsior (via r-universe)

July 9, 2026

Title Streamlines Excel File Automation

Version 0.0.1

Description Useful tools for automation of excel file data entry, especially transferring parts of one sheet into a sheet of another file.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

URL https://github.com/cbedwards-dfw/excelsior, https://cbedwards-dfw.github.io/excelsior/, https://framverse.r-universe.dev/excelsior,

BugReports https://github.com/cbedwards-dfw/excelsior/issues

Suggests janitor, knitr, rmarkdown, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Language en-US

Imports cli, dplyr, glue, openxlsx2, readxl, rlang, stringr, tibble, tidyr, tidyxl, utils, xldiff (>= 0.2.0), zoo

Remotes cbedwards-dfw/xldiff

VignetteBuilder knitr

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev

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

Date/Publication 2026-06-04 00:17:45 UTC

RemoteUrl https://github.com/cbedwards-dfw/excelsior

RemoteRef HEAD

RemoteSha 283cdb863e07d7fe5dba710f4d9ff51688607438

Contents

`as_numeric_smart` *Convert character vector to numeric with intelligent handling*

Description

Converts a character vector to numeric while intelligently handling common formatting that appears when reading mixed numeric and character cells from excel, like commas and percentages.

Usage

```
as_numeric_smart(vec)
```

Arguments

`vec`	A vector to convert to numeric

Value

A numeric vector with the following transformations applied:

- Commas are removed (e.g., "1,000" becomes 1000)
- Percentages are removed and those numbers are converted to proportions (e.g., "10%" becomes 0.1)
- Values that cannot be converted return NA

Examples

```
as_numeric_smart(c("1,000", "10%", "5.5", "text"))
# Returns: c(1000, 0.1, 5.5, NA)
```

clip_to_vec *Translate copied excel row/column into code to make an R vector*

Description

Function interacts with the computer clipboard. To use, start by copying a row or column in excel that you want coded as a vector in R. Run clip_to_vec() to change the system clipboard to the string of R code that defines an equivalent vector, then paste into R script.

Usage

```
clip_to_vec()
```

Value

Nothing

copy_columns *Copy columns from one Excel file to another with change tracking*

Description

Copies specified columns from an input Excel file to a workbook, highlighting changes compared to a master file and adding metadata about the transfer. `row_start` and `row_end` are most easily identified using `row_finder()`.

Usage

```
copy_columns(
  wb,
  to_file,
  from_file,
  sheet,
  columns,
  row_start,
  row_end,
  color_scheme,
  meta_cell,
  verbose = TRUE
)
```

Arguments

wb	An openxlsx2 workbook object to modify
to_file	Character. Filepath for the "original" reference file
from_file	Character. Filepath containing the workbook with data to copy FROM
sheet	Character. Sheet name to copy from/to
columns	Character vector. Excel column letters to copy (e.g., c("A", "B"))
row_start	Integer. Initial row number to copy
row_end	Integer. Final row number to copy
color_scheme	Character vector of length 2. Hex colors for "Changed" and "Unchanged" cells. Use the same general color with different shades.
meta_cell	Character. Excel cell reference to add transfer metadata (color, file, date).
verbose	Logical. Whether to print progress messages. Defaults to TRUE.

Value

Modified workbook object with copied data, change highlighting, and metadata

See Also

row_finder()

Examples

```
## Not run:
wb <- load_workbook("primary.xlsx)
wb <- copy_columns(wb,
                   to_file = "primary.xlsx",
                   from_file = "update.xlsx",
                   sheet = "Sheet1",
                   columns = c("A", "B"),
                   row_start = 2,
                   row_end = 100,
                   color_scheme = c("#82DE4E", "#D5F4C3"),
                   meta_cell =  "D1")

## End(Not run)
```

---

copy_section *Copy data section between locations in Excel workbook*

Description

Designed with updating in mind. Copies data from one location to another within an Excel workbook, with optional validation checks for matching reference rows/columns. This is a safe version of saying "Copy sheet C3:D4 of sheet 1 to E5:F6 of sheet 2". To support speed, this function takes as arguments the openxlsx2 workbook object to be updated, and a dataframe to update FROM.

Usage

```
copy_section(
  wb,
  from_df,
  from_address,
  to_address,
  sheet,
  check_row_offset = NULL,
  check_col_offset = NULL,
  numeric_flag = TRUE,
  debug_mode = FALSE
)
```

Arguments

wb	An openxlsx2 workbook object
from_df	A dataframe of the complete sheet that contains the data to copy *from*. Typically generated from `readxl::read_excel` with `col_names = FALSE` and `.name_repair = "unique_quiet`.
from_address	Character string specifying the source cell range (e.g., "A1:C3").
to_address	Character string specifying the destination cell range (e.g., "D1:F3")
sheet	Character string or numeric index specifying the target sheet
check_row_offset	Optional, integer offset identifying the row that should be used to validate that the from/to sections match. Positive values correspond to rows *above* the first row of the addresses specified.
check_col_offset	Optional, integer offset identifying the column that should be used to validate that the from/to sections match. Positive values correspond to columns *before* the first column of the addresses specified.
numeric_flag	Logical indicating whether data should be converted to numeric (default: TRUE)
debug_mode	Logical indicating whether to enable debug mode highlighting (default: FALSE)

Details

The key feature of this function is optional checking, in which the first column or row of the "from_db" is checked against a corresponding column or row of the "to_address" to make sure the entries are still appropriate.

As an example, when updating the TAMM template file with a new year's data, we have a rectangular section of locations and their values present in the TAMM and in the input template; we want to update the input template with the new TAMM values. We can code this with cell addresses, identifying that we want to copy from cells "D126:F130" of the TAMM to "D5:F9" of the current sheet of the input template. However, we could encounter problems if users have re-arranged either excel sheet. For this reason, we want to confirm that the location column (which is 1 column to the left of the leftmost cells we're copying) matches. So we provide "check_col_offset = 1" as an argument. This causes the function to check that "C123:C130" of the TAMM matches "C5:C9" of the current sheet of the input template.

Value

Modified workbook object with data copied

`find_fill_color` *Find fill color of specified cell*

Description

Find fill color of specified cell

Usage

```
find_fill_color(filepath, sheet, address)
```

Arguments

`filepath`	Excel file
`sheet`	Excel sheet
`address`	Cell address

Value

Excel rgb color (character vector of length 1). To use in R, add "#" to beginning.

Examples

```
## Not run:
find_fill_color(primary_file, "D-3", "A103")

## End(Not run)
```

`implant_df` *Directly implant data from dataframe to openxlsx2 workbook*

Description

Implants a dataframe into a specified cell range into an openxlsx2 workbook after validation checks have been completed.

Usage

```
implant_df(
  wb,
  new_dat,
  cell_range,
  sheet,
  numeric_flag = TRUE,
  debug_mode = FALSE
)
```

Arguments

wb	An openxlsx2 workbook object
new_dat	A dataframe containing the new data to be implanted
cell_range	Character string specifying the range of cells to copy into (e.g., "A1:C3")
sheet	Character string or numeric index specifying the sheet to copy into
numeric_flag	Logical indicating whether the data should be converted to numeric (default: TRUE)
debug_mode	Logical indicating whether to highlight updated cells in red (default: FALSE)

Details

When numeric_flag is TRUE, all columns are converted to numeric. When debug_mode is TRUE, updated cells are highlighted in blood red (#880808) and an alert message is printed.

Value

Modified workbook object with data implanted

---

read_excel_tiered_headers

Read an Excel file with multi-row headers

Description

Reads an Excel worksheet where column headers are spread across multiple rows, including merged cells. Header rows are combined into a single header string per column, with consecutive duplicate values collapsed (e.g. a superheader "TREATY" spanning columns that are individually labelled "TROLL", "NET", "SPORT" becomes "TREATY_TROLL", "TREATY_NET", "TREATY_SPORT").

Usage

```
read_excel_tiered_headers(
  path,
  sheet = 1,
  header_rows,
  pseudo_merged_rows = NULL,
  first_data_row = NULL,
  final_data_row = NULL,
  first_column = NULL,
  final_column = NULL,
  sep = "_",
  clean_names = TRUE
)
```

Arguments

path	Character. Path to the .xlsx file.
sheet	Integer or character atomic Sheet index to read. Default 1 to automatically handle cases with a single sheet.
header_rows	Integer vector. Row numbers (1-indexed, as they appear in the spreadsheet) that together form the column headers.
pseudo_merged_rows	Integer vector. Row numbers (1-indexed, as they appear in the spreadsheet) for header rows that include "false merged cells" – cells that look like they're merged because the first cell of a group has weird formatting that shifts the text far to the right. Warning: this can create funky names cases in which there are gaps columns in the merged row. This is NOT necessary if the excel cells are *actually* merged. Defaults to NULL.
first_data_row	Integer or NULL. Row number of the first data row. If NULL (default), set to max(header_rows) + 1.
final_data_row	Integer or NULL. Row number of the last data row. If NULL (default), all rows from first_data_row to the end of the sheet are returned.
first_column	Integer or NULL. Column number of the first column to read in. If NULL (default), uses default behavior of openxlsx2::read_xlsx().
final_column	Integer or NULL. Column number of the last column to read in. If NULL (default), uses default behavior of openxlsx2::read_xlsx().
sep	Character. Separator used when joining header parts from different rows. Default "_".
clean_names	Logical. Resolves issues in which column names are not unique. If TRUE (default), column names are passed through base::make.names() with unique = TRUE to ensure they are valid and non-duplicate R names.

Details

Merged cells are handled via openxlsx2::read_xlsx() with fill_merged_cells = TRUE, which propagates the anchor cell's value across the full span of the merge before any header combination is attempted.

Value

A data frame with one column per spreadsheet column and one row per data row. Column names are the combined headers derived from `header_rows`.

Examples

```
## Not run:
# Basic usage: rows 2-4 are headers, data starts at row 5
df <- read_excel_tiered_headers("data.xlsx")

# Custom header rows and separator
df <- read_excel_tiered_headers(
  path        = "data.xlsx",
  header_rows = c(1, 2),
  sep         = "."
)

# Read only a specific range of data rows
df <- read_excel_tiered_headers(
  path           = "data.xlsx",
  hearder_rows = c(2, 3, 4)
  first_data_row = 7,
  final_data_row = 100
)

## End(Not run)
```

`row_finder` *Find row number based on an "anchor cell"*

Description

Excel sheets are sometimes updated to add additional rows. This function find rows of interest based of off "anchor cells" – cells with known contents (e.g., column headers) in a known column. Using these instead of hard-coding row numbers can make automation less fragile to addition of rows.

Usage

```
row_finder(filepath, sheet, column, pattern, instance = 1, offset = 0)
```

Arguments

`filepath`	Character. Filepath to Excel file of interest
`sheet`	Character or numeric. Sheet name or index to search in
`column`	Character or numeric. Column that contains anchor cell as number (e.g., 3) or corresponding excel column label ("C").

pattern	Character. Pattern to identify anchor cell. Accepts regular expressions – see details.
instance	Numeric. Which match to use if multiple matches found? Defaults to 1.
offset	Numeric. Number to add to returned row number. Useful when pattern identifies a title row and you want the row below (1) or above (-1). Defaults to 0

Details

The pattern argument accepts regular expressions, which are more flexible than just exactly matching a string. A few tips:

- R treats \ specially – if present in the pattern, surround with square brackets to treat them as actual characters.
- In regular expressions, the ^ symbol is used to denote the start of a string, and the $ symbol is used to denote the end of a string.

As an example of this in practice, if we had an anchor cell that was the footnote "a/ Derived from vessel registrations and fish landing tickets.", we could use as a pattern any section of the text (e.g., pattern = "Derived from vessel registrations and fish"). But if we wanted to ensure that the first occurrence of "a/..." was used as our anchor (in case the footnote text changes), we could use the pattern "^a[/]", which in words means "the first letter is an a, and the second letter is literally a /".

Value

Numeric. Row number where pattern was found, adjusted by offset

Examples

```
## Not run:
row_finder("data.xlsx", "Sheet1", "A", "Total", instance = 1, offset = 1)
row_finder("data.xlsx", 1, 3, "Summary", offset = -1)

## End(Not run)
```

---

row_finder_by_color *Find anchor cell based on fill color*

Description

Useful for handling sheets that use color-filled rows as separators (e.g., STT table files).

Usage

```
row_finder_by_color(
  filepath,
  sheet,
  column = 1,
  color = "FF000000",
  instance = 1,
  offset = 0
)
```

Arguments

filepath	Character. Filepath to Excel file of interest
sheet	Character or numeric. Sheet name or index to search in
column	Character or numeric. Column that contains anchor cell as number (e.g., 3) or corresponding excel column label ("C").
color	Character vector. Color of anchor cell in hex, with or without preliminary "#". Defaults to black ("FF000000").
instance	Numeric. Which match to use if multiple matches found? Defaults to 1.
offset	Numeric. Number to add to returned row number. Useful when pattern identifies a title row and you want the row below (1) or above (-1). Defaults to 0

Value

Row number (numeric of length 1)

Examples

```
## Not run:
## find the row before the first column has a cell with black fill
row_finder_by_color("Appendix D.xlsx", sheet = "D-3", column = 1, offset = -1)

## End(Not run)
```

Index