

# Package: upstartr (via r-universe)

September 13, 2024

**Type** Package

**Title** Utilities Powering the Globe and Mail's Data Journalism Template

**Version** 0.1.2

**Maintainer** Tom Cardoso <tcardoso@globeandmail.com>

**Description** Core functions necessary for using The Globe and Mail's R data journalism template, 'startr', along with utilities for day-to-day data journalism tasks, such as reading and writing files, producing graphics and cleaning up datasets.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.6.3)

**Imports** here, stringr, readxl, magrittr, readr, purrr, ggplot2, glue, dplyr, librarian, openxlsx, knitr, beepr, tidytext, scales, rmarkdown, textclean, sf, tgamtheme, crayon

**Suggests** testthat (>= 3.0.0)

**Language** en-US

**URL** <https://github.com/globeandmail/upstartr>,  
<https://globeandmail.github.io/upstartr/>

**BugReports** <https://github.com/globeandmail/upstartr/issues>

**RoxygenNote** 7.2.3

**Config/testthat/edition** 3

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

**RemoteUrl** <https://github.com/globeandmail/upstartr>

**RemoteRef** HEAD

**RemoteSha** 963cd13163b29e241484abc52e116dfd31cab309

## Contents

begin_processing . . . . .	3
calc_index . . . . .	3
calc_mode . . . . .	4
clean_columns . . . . .	4
combine_csvs . . . . .	5
combine_excels . . . . .	5
convert_str_to_logical . . . . .	6
dir_data_cache . . . . .	7
dir_data_out . . . . .	7
dir_data_processed . . . . .	8
dir_data_raw . . . . .	8
dir_path . . . . .	9
dir_plots . . . . .	9
dir_reports . . . . .	10
dir_scrape . . . . .	10
dir_src . . . . .	11
end_processing . . . . .	11
initialize_starttr . . . . .	12
not.na . . . . .	13
not.null . . . . .	13
read_all_excel_sheets . . . . .	14
remove_non_utf8 . . . . .	14
render_notebook . . . . .	15
run_analyze . . . . .	15
run_config . . . . .	16
run_notebook . . . . .	16
run_process . . . . .	17
run_visualize . . . . .	17
scale_x_percent . . . . .	18
scale_y_percent . . . . .	18
simplify_string . . . . .	19
unaccent . . . . .	20
write_excel . . . . .	20
write_plot . . . . .	21
write_shp . . . . .	22
%not_in% . . . . .	22

## Index

23

---

begin_processing	<i>Runs the pre-processing step on a startr project.</i>
------------------	----------------------------------------------------------

---

### Description

The pre-processing step, run as part of `upstartr::run_process` during the `process.R` stage of a `startr` project, logs all variables currently in the global environment, which will then be removed during the post-processing step to keep the `startr` environment unpolluted.

### Usage

```
begin_processing(should_clean_processing_variables = TRUE)
```

### Arguments

`should_clean_processing_variables`  
 Either TRUE, FALSE, or pulled from the environment if set.

### Value

A list of all environment variables present before the function was run

---

calc_index	<i>Index values</i>
------------	---------------------

---

### Description

Index numeric vector to first value. By default, the index base will be 0, turning regular values into percentage change. In some cases, you may want to index to a different base, like 100, such as if you're looking at financial data.

### Usage

```
calc_index(m, base = 0)
```

### Arguments

`m` Numeric vector to index to first value.  
`base` Base to index against. (Default: 0)

### Value

An vector of indexed values.

**Examples**

```
calc_index(c(5, 2, 8, 17, 7, 3, 1, -4))
calc_index(c(5, 2, 8, 17, 7, 3, 1, -4), base = 100)
```

---

calc_mode	<i>Calculate mode</i>
-----------	-----------------------

---

**Description**

Calculates the mode of a given vector.

**Usage**

```
calc_mode(x)
```

**Arguments**

x Any kind of vector — numeric, character, logical.

**Value**

The mode(s) of that vector.

**Examples**

```
calc_mode(c(1, 1, 2, 3, 4))
calc_mode(c('the', 'quick', 'brown', 'fox', 'jumped', 'over', 'the', 'lazy', 'dog'))
calc_mode(c(TRUE, TRUE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE))
```

---

clean_columns	<i>Cleans up column names by forcing them into tidyverse style</i>
---------------	--------------------------------------------------------------------

---

**Description**

Zero-configuration function that takes unwieldy column names and coerces them into tidyverse-styled column names.

**Usage**

```
clean_columns(x)
```

**Arguments**

x A vector of column names.

**Value**

A character vector of column names.

**Examples**

```
clean_columns(c("Date of Purchase", "Item No.", "description", "",
               "Transaction at Jane's Counter?", "Auditing - Worth it?"))
```

---

combine_csvs	<i>Combine CSVs in a directory</i>
--------------	------------------------------------

---

**Description**

Given a directory (and, optionally, a pattern to search against), concatenate all CSV files into a single tibble.

**Usage**

```
combine_csvs(dir, pattern = "*.csv", ...)
```

**Arguments**

dir	Path to the directory to look at for files.
pattern	Pattern to use for detecting files. (Default: <code>'*.csv'</code> )
...	Parameters to pass to <code>readr::read_csv</code> .

**Value**

A tibble of concatenated data from multiple CSV files.

---

combine_excels	<i>Combine Excel files in a directory</i>
----------------	-------------------------------------------

---

**Description**

Given a directory (and, optionally, a pattern to search against), concatenate all Excel files into a single tibble.

**Usage**

```
combine_excels(dir, pattern = "*.xls[x]?", all_sheets = FALSE, ...)
```

**Arguments**

dir	Path to the directory to look at for files.
pattern	Pattern to use for detecting files. (Default: <code>'*.xls[x]?'</code> )
all_sheets	Should this function also concatenate all sheets within each Excel file into one long tibble? (Default: FALSE)
...	Parameters to pass to <code>readxl::read_excel</code> .

**Value**

A tibble of concatenated data from multiple Excel files.

---

convert\_str\_to\_logical

*Converts a character vector to logicals*

---

**Description**

Takes a character vector and converts it to logicals, optionally using a vector of patterns to match against for truthy and falsy values.

**Usage**

```
convert_str_to_logical(
  x,
  truthy = c("T", "TRUE", "Y", "YES"),
  falsy = c("F", "FALSE", "N", "NO")
)
```

**Arguments**

x	A character vector.
truthy	A vector of case-insensitive truthy values to turn into TRUE.
falsy	A vector of case-insensitive falsy values to turn into FALSE.

**Value**

A logical vector.

**Examples**

```
convert_str_to_logical(c('YES', 'Y', 'No', 'N', 'YES', 'yes', 'no', 'Yes', 'NO', 'Y', 'y'))
```

---

dir_data_cache	<i>Get path within cached data directory.</i>
----------------	-----------------------------------------------

---

**Description**

Constructs a path within starttr's data/cache/ directory.

**Usage**

```
dir_data_cache(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir_data_out	<i>Get path within disposable data outputs directory.</i>
--------------	-----------------------------------------------------------

---

**Description**

Constructs a path within starttr's data/out/ directory.

**Usage**

```
dir_data_out(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir\_data\_processed      *Get path within processed data directory.*

---

**Description**

Constructs a path within starttr's data/processed/ directory.

**Usage**

```
dir_data_processed(...)
```

**Arguments**

...      Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir\_data\_raw      *Get path within raw data directory.*

---

**Description**

Constructs a path within starttr's data/raw/ directory.

**Usage**

```
dir_data_raw(...)
```

**Arguments**

...      Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.



---

<code>dir_path</code>	<i>Construct an arbitrary path.</i>
-----------------------	-------------------------------------

---

**Description**

Convenience function that constructs a path. Wraps here: [:here](#).

**Usage**

```
dir_path(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

<code>dir_plots</code>	<i>Get path within plots directory.</i>
------------------------	-----------------------------------------

---

**Description**

Constructs a path within starttr's plots/ directory.

**Usage**

```
dir_plots(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir_reports	<i>Get path within reports directory.</i>
-------------	-------------------------------------------

---

**Description**

Constructs a path within starttr's reports/ directory.

**Usage**

```
dir_reports(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir_scrape	<i>Get path within scrape directory.</i>
------------	------------------------------------------

---

**Description**

Constructs a path within starttr's scrape/ directory.

**Usage**

```
dir_scrape(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

dir_src	<i>Get path within src directory</i>
---------	--------------------------------------

---

**Description**

Constructs a path within starttr's main R/ directory.

**Usage**

```
dir_src(...)
```

**Arguments**

... Any number of path strings, passed in the same fashion as here: [:here](#).

**Value**

A path string.

---

end_processing	<i>Runs the post-processing step on a starttr project.</i>
----------------	------------------------------------------------------------

---

**Description**

The post-processing step, run as part of `upstarttr::run_process` during the `process.R` stage of a starttr project, removes all variables saved by `upstarttr::begin_processing` and then beeps to announce it's finished.

**Usage**

```
end_processing(
  should_clean_processing_variables = TRUE,
  should_beep = TRUE,
  logged_vars = NULL
)
```

**Arguments**

`should_clean_processing_variables` Either TRUE, FALSE, or pulled from the environment if set.

`should_beep` Either TRUE, FALSE, or pulled from the environment if set.

`logged_vars` A list of variables that existed before the processing step began.

**Value**

No return value, called for side effects

---

initialize\_starttr      *Initialize starttr project*

---

## Description

Used to initialize a starttr template for analysis. Will enforce some starttr-required standards for analysis (such as removing scientific notation, setting timezones, and writing some project configs to ‘options’).

## Usage

```
initialize_starttr(
  author = "Firstname Lastname <firstlast@example.com>",
  title = "starttr",
  scipen = 999,
  timezone = "America/Toronto",
  should_render_notebook = FALSE,
  should_process_data = TRUE,
  should_timestamp_output_files = FALSE,
  should_clean_processing_variables = TRUE,
  should_beep = TRUE,
  set_minimal_graphics_theme = TRUE,
  packages = c()
)
```

## Arguments

author	Name and email of the starttr project author
title	Title of the starttr project
scipen	Which level of scientific precision to use. (Default: 999)
timezone	The timezone for analysis. (Default: ‘America/Toronto’)
should_render_notebook	Whether the RMarkdown notebook should be rendered. (Default: FALSE)
should_process_data	Whether starttr’s process step should be run. (Default: TRUE)
should_timestamp_output_files	Whether write_excel’s output files should be timestamped. (Default: FALSE)
should_clean_processing_variables	Whether processing variables should be cleaned from the environment after processing is complete. (Default: TRUE)
should_beep	Whether starttr should beep after tasks like processing or knitting RMarkdown notebooks. (Default: TRUE)
set_minimal_graphics_theme	Whether the minimal graphics theme should be used. (Default: TRUE)



**Value**

Elements that aren't NULL

**Examples**

```
not.null(list(1, NULL, 2, NULL))
```

---

`read_all_excel_sheets` *Combine all sheets in an Excel file*

---

**Description**

Reads all sheets in a single Excel file using `readxl::read_excel` and concatenates them into a single, long tibble.

**Usage**

```
read_all_excel_sheets(filepath, ...)
```

**Arguments**

`filepath` Path to the Excel file.  
`...` Parameters to pass to `readxl::read_excel`.

**Value**

A tibble data concatenated from all sheets in an Excel file.

---

`remove_non_utf8` *Removes non-UTF-8 characters*

---

**Description**

Removes non-UTF-8 characters in a given character vector.

**Usage**

```
remove_non_utf8(x)
```

**Arguments**

`x` A character vector.

**Value**

A character vector of strings without non-UTF-8 characters.

**Examples**

```
non_utf8 <- 'fa\xE7ile'  
Encoding(non_utf8) <- 'latin1'  
remove_non_utf8(non_utf8)
```

---

render_notebook	<i>Renders out an RMarkdown notebook.</i>
-----------------	-------------------------------------------

---

**Description**

Renders an RMarkdown notebook using `upstart::render_notebook` and then beeps.

**Usage**

```
render_notebook(notebook_file, output_dir = dir_reports())
```

**Arguments**

`notebook_file` The path for the RMarkdown notebook you're rendering.  
`output_dir` The directory to write the outputs to.

**Value**

No return value, called for side effects

---

run_analyze	<i>Runs the analysis step for a startr project.</i>
-------------	-----------------------------------------------------

---

**Description**

Sources `analyze.R`.

**Usage**

```
run_analyze()
```

**Value**

No return value, called for side effects

---

run_config	<i>Configures an existing starttr project</i>
------------	-----------------------------------------------

---

**Description**

Sources config.R and functions.R in turn.

**Usage**

```
run_config()
```

**Value**

No return value, called for side effects

---

run_notebook	<i>Runs the notebook rendering step for a starttr project.</i>
--------------	----------------------------------------------------------------

---

**Description**

Renders an RMarkdown notebook using upstarttr: [render\\_notebook](#) and then beeps.

**Usage**

```
run_notebook(  
  filename = "notebook.Rmd",  
  should_beep = TRUE,  
  should_render_notebook = TRUE  
)
```

**Arguments**

filename	The filename for the RMarkdown notebook you want to render.
should_beep	Either TRUE, FALSE, or pulled from the environment if set.
should_render_notebook	Either TRUE, FALSE, or pulled from the environment if set.

**Value**

No return value, called for side effects



---

run_process	<i>Runs the processing step on a starttr project.</i>
-------------	-------------------------------------------------------

---

**Description**

Runs the pre-processing step (see `upstarttr::begin_processing` for details), then sources `process.R`, then runs the post-processing step (see `upstarttr::end_processing` for details).

**Usage**

```
run_process(should_process_data = TRUE)
```

**Arguments**

`should_process_data`  
Either TRUE, FALSE, or pulled from the environment if set.

**Value**

No return value, called for side effects

---

run_visualize	<i>Runs the visualization step for a starttr project.</i>
---------------	-----------------------------------------------------------

---

**Description**

Sources `visualize.R`.

**Usage**

```
run_visualize()
```

**Value**

No return value, called for side effects

---

scale_x_percent	<i>Create a continuous x-axis scale using percentages</i>
-----------------	-----------------------------------------------------------

---

**Description**

Convenience function to return a `scale_x_continuous` function using percentage labels.

**Usage**

```
scale_x_percent(...)
```

**Arguments**

... All your usual continuous x-axis scale parameters.

**Value**

A scale object to be consumed by `ggplot2`.

---

scale_y_percent	<i>Create a continuous y-axis scale using percentages</i>
-----------------	-----------------------------------------------------------

---

**Description**

Convenience function to return a `scale_y_continuous` function using percentage labels.

**Usage**

```
scale_y_percent(...)
```

**Arguments**

... All your usual continuous y-axis scale parameters.

**Value**

A scale object to be consumed by `ggplot2`.

---

simplify_string	<i>Simplifies strings for analysis</i>
-----------------	----------------------------------------

---

### Description

Takes a character vector and "simplifies" it by uppercasing, removing most non-alphabetic (or alphanumeric) characters, removing accents, forcing UTF-8 encoding, removing excess spaces, and optionally removing stop words. Useful in cases where you have two large vector of person or business names you need to compare, but where misspellings may be common.

### Usage

```
simplify_string(  
  x,  
  alpha = TRUE,  
  digits = FALSE,  
  unaccent = TRUE,  
  utf8_only = TRUE,  
  case = "upper",  
  trim = TRUE,  
  stopwords = NA  
)
```

### Arguments

x	A character vector.
alpha	Should alphabetic characters be included in the cleaned up string? (Default: TRUE)
digits	Should digits be included in the cleaned up string? (Default: FALSE)
unaccent	Should characters be de-accented? (Default: TRUE)
utf8_only	Should characters be UTF-8 only? (Default: TRUE)
case	What casing should characters use? Can be one of 'upper', 'lower', 'sentence', 'title', or 'keep' for the existing casing (Default: 'upper')
trim	Should strings be trimmed of excess spaces? (Default: TRUE)
stopwords	An optional vector of stop words to be removed.

### Value

A character vector of simplified strings.

### Examples

```
simplify_string(c('J. Jonah Jameson', 'j jonah jameson',  
  'j jonah 123 jameson', 'J Jónah Jameson...'))  
simplify_string(c('123 Business Inc.', '123 business incorporated',  
  '123 ... Business ... Inc.'), digits = TRUE, stopwords = c('INC', 'INCORPORATED'))
```

---

unaccent	<i>De-accents strings</i>
----------	---------------------------

---

### Description

Replace accented characters with their non-accented versions. Useful when dealing with languages like French, Spanish or Portuguese, where accents can lead to compatibility issues during data analysis.

### Usage

```
unaccent(x, remove.nonconverted = FALSE, ...)
```

### Arguments

x	A character vector.
remove.nonconverted	Should the function remove unmapped encodings? (Default: FALSE)
...	Parameters passed to <code>textclean::replace_non_ascii</code>

### Value

A character vector of strings without accents.

### Examples

```
unaccent('façile')  
unaccent('Montréal')
```

---

write_excel	<i>Write out an Excel file with minimal configuration</i>
-------------	-----------------------------------------------------------

---

### Description

Takes a tibble or dataframe variable and saves it out as an Excel file using the variable name as the filename.

### Usage

```
write_excel(  
  variable,  
  output_dir = dir_data_out(),  
  should_timestamp_output_files = FALSE  
)
```

**Arguments**

variable	A tibble or dataframe object.
output_dir	The directory to save the file out to.
should_timestamp_output_files	Either TRUE, FALSE, or pulled from the environment if set.

**Value**

No return value, called for side effects

---

write_plot	<i>Write out a ggplot2 graphic with minimal configuration</i>
------------	---------------------------------------------------------------

---

**Description**

Takes a ggplot2 object and writes it to disk via `ggplot2::ggsave` using the variable name as the filename.

**Usage**

```
write_plot(variable, format = "png", output_dir = dir_plots(), ...)
```

**Arguments**

variable	A tibble or dataframe object.
format	The desired format for the plot, be it 'png', 'pdf', etc. Accepts formats you'd pass to <code>ggplot2::ggsave</code> 's 'device' parameter.
output_dir	The directory to save the plot out to.
...	Other settings to pass to <code>ggsave</code> , such as format, width, height or dpi.

**Value**

No return value, called for side effects

---

<code>write_shp</code>	<i>Write a shapefile to disk</i>
------------------------	----------------------------------

---

**Description**

Utility function that wraps `sf::st_write`, but first removes a previous version of the shapefile if it exists (by default, `sf::st_write` will throw an error.)

**Usage**

```
write_shp(shp, path, ...)
```

**Arguments**

<code>shp</code>	A spatial object.
<code>path</code>	The desired filepath for the shapefile.
<code>...</code>	Other settings to pass to <code>st_write</code> , such as format, width, height or dpi.

**Value**

No return value, called for side effects

---

<i>%not_in%</i>	<i>Opposite of %in%</i>
-----------------	-------------------------

---

**Description**

Given vectors A and B, returns only the entities from vector A that don't occur in vector B.

**Usage**

```
x %not_in% table
```

**Arguments**

<code>x</code>	The vector you want to check.
<code>table</code>	Table in which to do lookups against x.

**Value**

Same form of return as *%in%* — except it will return only elements on the lhs that aren't present on the rhs

**Examples**

```
c(1, 2, 3, 4, 5) %not_in% c(4, 5, 6, 7, 8)
```

# Index

[%not\\_in%](#), [22](#)

[begin\\_processing](#), [3](#), [11](#), [17](#)

[calc\\_index](#), [3](#)  
[calc\\_mode](#), [4](#)  
[clean\\_columns](#), [4](#)  
[combine\\_csvs](#), [5](#)  
[combine\\_excels](#), [5](#)  
[convert\\_str\\_to\\_logical](#), [6](#)

[dir\\_data\\_cache](#), [7](#)  
[dir\\_data\\_out](#), [7](#)  
[dir\\_data\\_processed](#), [8](#)  
[dir\\_data\\_raw](#), [8](#)  
[dir\\_path](#), [9](#)  
[dir\\_plots](#), [9](#)  
[dir\\_reports](#), [10](#)  
[dir\\_scrape](#), [10](#)  
[dir\\_src](#), [11](#)

[end\\_processing](#), [11](#), [17](#)

[ggsave](#), [21](#)

[here](#), [7–11](#)

[initialize\\_startr](#), [12](#)

[not.na](#), [13](#)  
[not.null](#), [13](#)

[read\\_all\\_excel\\_sheets](#), [14](#)  
[read\\_csv](#), [5](#)  
[read\\_excel](#), [6](#), [14](#)  
[remove\\_non\\_utf8](#), [14](#)  
[render\\_notebook](#), [15](#), [15](#), [16](#)  
[replace\\_non\\_ascii](#), [20](#)  
[run\\_analyze](#), [15](#)  
[run\\_config](#), [16](#)  
[run\\_notebook](#), [16](#)

[run\\_process](#), [3](#), [11](#), [17](#)  
[run\\_visualize](#), [17](#)

[scale\\_x\\_percent](#), [18](#)  
[scale\\_y\\_percent](#), [18](#)  
[simplify\\_string](#), [19](#)  
[st\\_write](#), [22](#)

[unaccent](#), [20](#)

[write\\_excel](#), [20](#)  
[write\\_plot](#), [21](#)  
[write\\_shp](#), [22](#)