Package ‘k5’

March 11, 2024

**Title**  
Kiernan Nicholls Miscellaneous

**Version**  
0.2.1

**Description**  
Quality of life functions for interactive programming.  
Shortcuts for common combinations of functions or different default  
arguments. Not to be used in production level scripts, but useful for  
exploring and quickly manipulating data for easy analysis. Also  
imports a variety of packages to facilitate the installation of those  
imported packages on the host machine.

**License**  
GPL-3

**URL**  
https://k5cents.github.io/k5/, https://github.com/k5cents/k5

**BugReports**  
https://github.com/k5cents/k5/issues

**Depends**  
R (>= 2.10)

**Imports**  
clipr (>= 0.8.0), dplyr (>= 1.1.3), fs (>= 1.6.3), ggplot2 (>= 3.4.4), glue (>= 1.6.2), lubridate (>= 1.9.3), magrittr (>= 2.0.3), purrr (>= 1.0.2), readr (>= 2.1.4), rlang (>= 1.1.1), stringr (>= 1.5.0), tibble (>= 3.2.1), usethis (>= 2.2.2), utils

**Suggests**  
covr (>= 3.6.1), crayon (>= 1.5.2), gluedown (>= 1.0.6), here (>= 1.0.1), http (>= 1.4.4), janitor (>= 2.1.0), knitr (>= 1.41), pacman (>= 0.5.1), readxl (>= 1.4.1), rvest (>= 1.0.3), scales (>= 1.2.1), testthat (>= 3.1.6)

**Encoding**  
UTF-8

**LazyData**  
true

**RoxygenNote**  
7.2.3

**NeedsCompilation**  
no

**Author**  
Kiernan Nicholls [aut, cre, cph]  
(<https://orcid.org/0000-0002-9229-7897>)

**Maintainer**  
Kiernan Nicholls <k5cents@gmail.com>

**Repository**  
CRAN

**Date/Publication**  
2024-03-11 07:50:02 UTC
**R topics documented:**

- `contract_convert` ................................................................. 2
- `copy_last` ............................................................................. 3
- `count2` .................................................................................. 4
- `count_diff` ............................................................................ 4
- `count_in` ............................................................................... 5
- `count_na` ............................................................................... 6
- `count_out` ............................................................................. 7
- `file_age` ................................................................................ 8
- `file_encoding` ........................................................................ 8
- `filter_rx` ............................................................................... 9
- `gaa` ....................................................................................... 9
- `last_value` ............................................................................ 10
- `load_my_packages` .............................................................. 10
- `na_in` .................................................................................... 11
- `na_out` .................................................................................. 11
- `na_rep` .................................................................................. 12
- `print_all` .............................................................................. 13
- `prop_distinct` ....................................................................... 13
- `prop_in` ............................................................................... 14
- `prop_na` ............................................................................... 15
- `prop_out` ............................................................................. 16
- `read_delim_clip` ................................................................... 17
- `read_delim_dumb` ............................................................... 17
- `var_missing` ......................................................................... 18
- `view_firefox` ........................................................................ 18
- `view_last` .............................................................................. 19
- `what_in` ............................................................................... 19
- `what_out` ............................................................................ 20
- `word_count` .......................................................................... 21
- `write_delim_clip` ................................................................. 21
- `write_last` ............................................................................ 22
- `%out%` .................................................................................. 23

---

**Index**

<table>
<thead>
<tr>
<th><code>contract_convert</code></th>
<th><strong>Convert contract names to factor intervals</strong></th>
</tr>
</thead>
</table>

**Description**

Can perform one of three **rough** conversions:

1. For interval contracts (e.g., "220 - 229", "9% or more", etc.), convert the character strings to proper interval notation.

2. For contracts with multiple discrete outcomes (e.g., Candidate names), convert the character vector to simple factors.
3. For markets with a single binary question (e.g., "Will the Democrats have a brokered convention in 2020?"), contracts returned are always "Yes" which is converted to TRUE.

Usage

```r
contract_convert(x, decimal = FALSE)
```

Arguments

- `x`: A character vector of contract names.
- `decimal`: Should percentages be converted to decimals?

Value

A interval factor, unique factor, or logical vector.

---

**copy_last**

*Copy the last object to the clipboard*

Description

Use `clipr::write_clip()` to write the last value as a character vector to the system clipboard.

Usage

```r
copy_last(x = .Last.value)
```

Arguments

- `x`: The object to view, usually left as `base::Last.value`.

Details

The value of the internal evaluation of a top-level R expression is always assigned to `.Last.value` before further processing (e.g., printing).

Value

The same `.Last.value` as before copied, invisibly.
count2: Count the way you want

Description
A wrapper around dplyr::count() with sort set to TRUE by default and the an additional column created by default containing the proportional fraction each observation makes of the whole.

Usage
```r
count2(x, ..., wt = NULL, sort = TRUE, prop = TRUE, sum = NULL)
```
```r
count_vec(x, sort = TRUE, prop = TRUE, sum = NULL)
```

Arguments
- `x`: A data frame.
- `...`: Variables to group by.
- `wt`: Frequency weights.
- `sort`: If TRUE, will show the largest groups at the top.
- `prop`: If TRUE, compute the fraction of marginal table.
- `sum`: Column to replace with a cumulative sum (n, p, or np).

Value
A tibble of element counts

Examples
```r
count2(iris, Species)
```

count_diff: Count set difference

Description
Find the length of the set of difference between x and y vectors.

Usage
```r
count_diff(x, y, ignore.case = FALSE)
```
**count_in**

**Arguments**

- **x**  
  A vector to check.
- **y**  
  A vector to compare against.
- **ignore.case**  
  logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

**Details**

\[
\text{sum}(x \%out\% y)
\]

**Value**

The number of *unique* values of x not in y.

**See Also**

Other counting wrappers: \texttt{count_in()}, \texttt{count_na()}, \texttt{count_out()}, \texttt{na_in()}, \texttt{na_out()}, \texttt{na_rep()}, \texttt{prop_distinct()}, \texttt{prop_in()}, \texttt{prop_na()}, \texttt{prop_out()}, \texttt{what_in()}, \texttt{what_out()}

**Examples**

```
# only unique values are checked
count_diff(c("VT", "NH", "ZZ", "ZZ", "ME"), state.abb)
```

```
count_in  Count in

Description

Count the total values of x that are \texttt{%in%} the vector y.

Usage

\texttt{count_in(x, y, na.rm = TRUE, ignore.case = FALSE)}

Arguments

- **x**  
  A vector to check.
- **y**  
  A vector to compare against.
- **na.rm**  
  logical; Should NA be ignored?
- **ignore.case**  
  logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

Details

\[
\text{sum}(x \%out\% y)
\]
Value

The sum of x present in y.

See Also

Other counting wrappers: `count_diff()`, `count_na()`, `count_out()`, `na_in()`, `na_out()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()`

Examples

count_in(c("VT", "NH", "ZZ", "ME"), state.abb)

count_na(c("VT", "NH", NA, "ME"))

---

Description

Count the total values of x that are NA.

Usage

count_na(x)

Arguments

x A vector to check.

Details

sum(is.na(x))

Value

The sum of x that are NA

See Also

Other counting wrappers: `count_diff()`, `count_in()`, `count_out()`, `na_in()`, `na_out()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()`

Examples

count_na(c("VT", "NH", NA, "ME"))
Description

Count the total values of \( x \) that are %out% of the vector \( y \).

Usage

```r
count_out(x, y, na.rm = TRUE, ignore.case = FALSE)
```

Arguments

- **x**: A vector to check.
- **y**: A vector to compare against.
- **na.rm**: logical; Should NA be ignored?
- **ignore.case**: logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

Details

```r
sum(x %out% y)
```

Value

The sum of \( x \) absent in \( y \).

See Also

Other counting wrappers: `count_diff()`, `count_in()`, `count_na()`, `na_in()`, `na_out()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()`

Examples

```r
count_out(c("VT", "NH", "ZZ", "ME"), state.abb)
```
file_age  

Description
The period of time since a system file was modified.

Usage
file_age(...)

Arguments
...  Arguments passed to file.info(), namely character vectors containing file paths. Tilde-expansion is done: see path.expand().

Value
A Period class object.

Examples
file_age(system.file("README.md", package = "campfin"))

file_encoding  

Description
Call the file command line tool with option -i.

Usage
file_encoding(path)

Arguments
path  A local file path or glob to check.

Value
A tibble of file encoding.
filter_rx  

Filter a data frame by a regular expression

Description
A shortcut for dat %>% filter(str_detect(column, "\d")).

Usage
filter_rx(dat, col, pattern, ...)

Arguments
- dat: A data frame with a character column to filter.
- col: The column containing a character vector to input.
- pattern: Pattern to look for.
- ...: Additional arguments passed to stringr::str_detect().

Value
A subset of rows from dat.

gaa  

GAA Team Abbreviations by Season and Team ID

Description
GAA Team Abbreviations by Season and Team ID

Usage
gaa

Format
A data frame with 74 rows and 3 variables:

- seasonId: The fantasy season integer
- teamId: The team ID integer
- abbrev: The normalized owner abbreviation for that year...
### last_value

*Return the last value*

**Description**

A function shortcut for accessing `.Last.value`.

**Usage**

```r
def last_value(x = .Last.value)
```

**Arguments**

- `x` The object to return, usually left as `base::.Last.value`.

**Details**

The value of the internal evaluation of a top-level R expression is always assigned to `.Last.value` (in package:base) before further processing (e.g., printing).

**Value**

The same `.Last.value` as before viewing, invisibly.

---

### load_my_packages

*Save and load packages from file*

**Description**

Save and load packages from file

**Usage**

```r
def load_my_packages(path = NULL)
```

```r
def save_my_packages(x = NULL, path = tempfile())
```

**Arguments**

- `path` The path to a text file containing one package per line. If NULL (default), then the default list is read from k5/inst/PACKAGES.
- `x` A character vector of package names to save. If NULL (default), use all currently attached packages.

**Value**

The list of packages, invisibly.
**na_in**

---

**Remove in**

**Description**

Set NA for the values of x that are `%in%` the vector y.

**Usage**

```r
na_in(x, y, ignore.case = FALSE)
```

**Arguments**

- **x**: A vector to check.
- **y**: A vector to compare against.
- **ignore.case**: logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

**Value**

The vector x missing any values in y.

**See Also**

Other counting wrappers: `count_diff()`, `count_in()`, `count_na()`, `count_out()`, `na_out()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()`

**Examples**

```r
na_in(c("VT", "NH", "ZZ", "ME"), state.abb)
na_in(1:10, seq(1, 10, 2))
```

---

**na_out**

---

**Remove out**

**Description**

Set NA for the values of x that are `%out%` of the vector y.

**Usage**

```r
na_out(x, y, ignore.case = FALSE)
```
Arguments

- **x**: A vector to check.
- **y**: A vector to compare against.
- **ignore.case**: logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

Value

The vector x missing any values not in y.

See Also

Other counting wrappers: `count_diff()`, `count_in()`, `count_na()`, `count_out()`, `na_in()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()`

Examples

```r
na_out(c("VT", "NH", "ZZ", "ME"), state.abb)
na_out(1:10, seq(1, 10, 2))
```

Description

Set NA for the values of x that contain a single repeating character and no other characters.

Usage

```r
na_rep(x, n = 0)
```

Arguments

- **x**: A vector to check.
- **n**: The minimum number times a character must repeat. If 0, the default, then any string of one character will be replaced with NA. If greater than 0, the string must contain greater than n number of repetitions.

Details

Uses the regular expression "^(.)\1+$".

Value

The vector x with NA replacing repeating character values.
print_all

See Also
Other counting wrappers: `count_diff()`, `count_in()`, `count_na()`, `count_out()`, `na_in()`, `na_out()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `prop_out()`, `what_in()`, `what_out()

Examples
`na_rep(c("VT", "NH", "ZZ", "ME"))`

print_all  
Print all rows of elements

Description
Print up to the `getOption("max.print")` and ask the user if they want to print more than that. This is most useful when printing tibbles with more than 10 rows but less than `getOption("max.print")`.

Usage
`print_all(x, ask = TRUE)`

Arguments

- `x` Object to print, typically a data frame or vector.
- `ask` If the length of `x` exceeds `getOption("max.print")`, should the user be prompted to confirm their intention to print everything. If `FALSE`, the maximum is printed without double checking: this can be extremely slow. The 'usethis' package must be installed for interactive confirmation.

Value
The object `x` (invisibly)

prop_distinct  
Proportion missing

Description
Find the proportion of values of `x` that are distinct.

Usage
`prop_distinct(x)`

Arguments

- `x` A vector to check.
Details

\[
\frac{\text{length(unique}(x))}{\text{length}(x)}
\]

Value

The ratio of distinct values \( x \) to total values of \( x \).

See Also

Other counting wrappers: \texttt{count_diff()}, \texttt{count_in()}, \texttt{count_na()}, \texttt{count_out()}, \texttt{na_in()}, \texttt{na_out()}, \texttt{na_rep()}, \texttt{prop_in()}, \texttt{prop_na()}, \texttt{prop_out()}, \texttt{what_in()}, \texttt{what_out()}

Examples

\[
\text{prop_distinct(c("VT", "VT", NA, "ME"))}
\]

---

\textbf{Description}

Find the proportion of values of \( x \) that are \texttt{in} the vector \( y \).

Usage

\[
\text{prop_in}(x, y, \text{na.rm} = \text{TRUE}, \text{ignore.case} = \text{FALSE})
\]

Arguments

\begin{itemize}
\item \textit{x} \hfill A vector to check.
\item \textit{y} \hfill A vector to compare against.
\item \textit{na.rm} \hfill logical; Should NA be ignored?
\item \textit{ignore.case} \hfill logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.
\end{itemize}

Details

\[
\text{mean}(x \texttt{in } y)
\]

Value

The proportion of \( x \) present in \( y \).

See Also

Other counting wrappers: \texttt{count_diff()}, \texttt{count_in()}, \texttt{count_na()}, \texttt{count_out()}, \texttt{na_in()}, \texttt{na_out()}, \texttt{na_rep()}, \texttt{prop_distinct()}, \texttt{prop_na()}, \texttt{prop_out()}, \texttt{what_in()}, \texttt{what_out()}
prop_na

Examples

prop_in(c("VT", "NH", "ZZ", "ME"), state.abb)

<table>
<thead>
<tr>
<th>prop_na</th>
<th>Proportion missing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Description

Find the proportion of values of \( x \) that are NA.

Usage

prop_na(x)

Arguments

\( x \)  
A vector to check.

Details

\[
\text{mean(is.na}(x)\text{)}
\]

Value

The proportion of values of \( x \) that are NA.

See Also

Other counting wrappers: \text{count_diff()}, \text{count_in()}, \text{count_na()}, \text{count_out()}, \text{na_in()}, \text{na_out()}, \text{na_rep()}, \text{prop_distinct()}, \text{prop_in()}, \text{prop_out()}, \text{what_in()}, \text{what_out()}

Examples

prop_na(c("VT", "NH", NA, "ME"))
**prop_out**

**Propportion out**

**Description**

Find the proportion of values of \( x \) that are %out% of the vector \( y \).

**Usage**

```r
prop_out(x, y, na.rm = TRUE, ignore.case = FALSE)
```

**Arguments**

- **x** A vector to check.
- **y** A vector to compare against.
- **na.rm** logical; Should NA be ignored?
- **ignore.case** logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

**Details**

```r
mean(x %out% y)
```

**Value**

The proportion of \( x \) absent in \( y \).

**See Also**

Other counting wrappers: `count_diff()`, `count_in()`, `count_na()`, `count_out()`, `na_in()`, `na_out()`, `na_rep()`, `prop_distinct()`, `prop_in()`, `prop_na()`, `what_in()`, `what_out()`

**Examples**

```r
c(prop_out(c("VT", "NH", "ZZ", "ME"), state.abb))
```
**read_delim_clip**  
*Read a table from the clipboard*

**Description**  
Use `readr::read_delim()` on a string copied to the clipboard. Defaults to tab separator like given when copying cells from spreadsheets.

**Usage**  
```r  
read_delim_clip(delim = "\t", ...)  
```

**Arguments**  
- `delim`  
  Single character used to separate fields within a record.  
- `...`  
  Additional arguments passed to `readr::read_delim()`.

**Value**  
A data frame read from the clipboard.

**read_delim_dumb**  
*Read a text file without column guessing*

**Description**  
Use `readr::read_delim()` without specifying any column types. All columns are treated as character strings.

**Usage**  
```r  
read_delim_dumb(file, delim = c(",", "\t", "|"), ...)  
read_csv_dumb(file, ...)  
read_tsv_dumb(file, ...)  
```

**Arguments**  
- `file`  
  Either a path to a file, a connection, or literal data.  
- `delim`  
  Single character used to separate fields within a record.  
- `...`  
  Additional arguments passed to `readr::read_delim()`.

**Value**  
A tibble data frame read from the file.
var_missing  

Apply a statistic function to all variables  

Description

Apply either `count_na()` or `dplyr::n_distinct()` to every column of a data frame and return the count and share of total values (either proportion missing or proportion distinct).

Usage

```r
var_missing(df)
var_distinct(df)
```

Arguments

- `df`: A data frame to glimpse.

Value

Invisibly, a table of statistics by column of a data frame.

Examples

```r
var_missing(dplyr::storms)
var_distinct(dplyr::storms)
```

---

view_firefox  

View an HTML document in Firefox  

Description

Take an XML document object, write to an HTML file, and open in Firefox.

Usage

```r
view_firefox(html)
```

Arguments

- `html`: An object which has the class `xml_document`, often from `rvest`.

Value

The `html` object, invisibly.
**view_last**

**View the last object**

---

**Description**

Invoke a spreadsheet-style data viewer on a matrix-like R object. In a non-interactive session, the object is returned invisibly and nothing is "viewed".

**Usage**

```r
view_last(x = .Last.value)
```

**Arguments**

- `x` 
  The object to view, usually left as `base::.Last.value`.

**Details**

The value of the internal evaluation of a top-level R expression is always assigned to `.Last.value` before further processing (e.g., printing).

**Value**

The same `.Last.value` as before viewing, invisibly.

---

**what_in**

**Which in**

---

**Description**

Return the values of `x` that are `%in%` of the vector `y`.

**Usage**

```r
what_in(x, y, ignore.case = FALSE)
```

**Arguments**

- `x` 
  A vector to check.
- `y` 
  A vector to compare against.
- `ignore.case` 
  logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

**Details**

`x[which(x %in% y)]`
Value
The elements of x that are %in% y.

See Also
Other counting wrappers: count_diff(), count_in(), count_na(), count_out(), na_in(), na_out(), na_rep(), prop_distinct(), prop_in(), prop_na(), prop_out(), what_out()

Examples
what_in(c("VT", "DC", NA), state.abb)

what_out
---

Which out

Description
Return the values of x that are %out% of the vector y.

Usage
what_out(x, y, na.rm = TRUE, ignore.case = FALSE)

Arguments
- x: A vector to check.
- y: A vector to compare against.
- na.rm: logical; Should NA be ignored?
- ignore.case: logical; if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

Details
x[which(x %out% y)]

Value
The elements of x that are %out% y.

See Also
Other counting wrappers: count_diff(), count_in(), count_na(), count_out(), na_in(), na_out(), na_rep(), prop_distinct(), prop_in(), prop_na(), prop_out(), what_in()

Examples
what_out(c("VT", "DC", NA), state.abb)
**word_count**

Count file words, lines, and bytes

**Description**

Invoke system tool to print newline, word, and byte counts for each file.

**Usage**

```r
word_count(path, count = ""
```

**Arguments**

- **path** Character vector of file paths.
- **count** The type of element to count, see details.

**Details**

One of five options or an empty string (default):

1. "lines" for newline characters (separating lines).
2. "words" for words separated by white space.
3. "chars" for individual characters.
4. "bytes" for total bytes, differs with multibyte characters.
5. "max" for the maximum display width of longest line.

**Value**

A data frame of counts by file.

---

**write_delim_clip**

Write a table from the clipboard

**Description**

Use `readr::format_delim()` on a data frame to copy a string to the clipboard. Defaults to tab separator like given when copying cells from spreadsheets.

**Usage**

```r
write_delim_clip(x, delim = "\t", ...)
```
Arguments

- **file**  
  File or connection to write to.

- **x**  
  The object to write, usually left as `base::.Last.value`.

- **...**  
  Additional arguments passed to the writing function (see Details).

Value

Invisibly, the input data frame.

---

**write_last**  
*Write the last value to disk*

Description

The value of the internal evaluation of a top-level R expression is always assigned to `.Last.value` before further processing (e.g., printing).

Usage

```r
write_last(file = tempfile(), x = .Last.value, ...)

save_last(file = tempfile(), x = .Last.value, ...)
```

Arguments

- **file**  
  File or connection to write to.

- **x**  
  The object to write, usually left as `base::.Last.value`.

- **...**  
  Additional arguments passed to the writing function (see Details).

Details

Four types of files are written, based on object class:

1. For data frames, a tab-separated file via `readr::write_tsv()`.
2. For vectors, a newline-separated file via `readr::write_lines()`.
3. For ggplots, a raster image (by default) via `ggplot2::ggsave()`.
4. For other objects, an uncompressed data file via `readr::write_rds()`.

Value

The created file path, invisibly.
%out% is an inverted version of the infix %in% operator.

Usage

\[ x \ %out\% \ table \]

Arguments

- \( x \) vector: the values to be matched. Long vectors are supported.
- \( table \) vector or NULL: the values to be matched against.

Details

%out% is currently defined as

\[ %out% <- function(x, table) match(x, table, nomatch = 0) == 0 \]

Value

logical; if \( x \) is not present in \( table \)

Examples

\[ c("A", "B", "3") \ %out\% LETTERS \]
Index

* counting wrappers
  count_diff, 4
count_in, 5
count_na, 6
count_out, 7
na_in, 11
na_out, 11
na_rep, 12
prop_distinct, 13
prop_in, 14
prop_na, 15
prop_out, 16
what_in, 19
what_out, 20

datasets
  gaa, 9
  .Last.value, 10
%out%, 23
base::.Last.value, 3, 10, 19, 22
clipr::write_clip(), 3
contract_convert, 2
copy_last, 3
count2, 4
count_diff, 4, 6, 7, 11–16, 20
count_in, 5, 5, 6, 7, 11–16, 20
count_na, 5, 6, 6, 7, 11–16, 20
count_na(), 18
count_out, 5, 6, 7, 11–16, 20
count_vec(count2), 4
dplyr::count(), 4
dplyr::n_distinct(), 18
file.info(), 8
file_age, 8
file_encoding, 8
filter_rx, 9
gaa, 9
ggplot2::ggsave(), 22
last_value, 10
load_my_packages, 10
na_in, 5–7, 11, 12–16, 20
na_out, 5–7, 11, 11, 13–16, 20
na_rep, 5–7, 11, 12, 12, 14–16, 20
path.expand(), 8
print_all, 13
prop_distinct, 5–7, 11–13, 13, 14–16, 20
prop_in, 5–7, 11–14, 14, 15, 16, 20
prop_na, 5–7, 11–14, 15, 16, 20
prop_out, 5–7, 11–15, 16, 20
read_csv_dumb (read_delim_dumb), 17
read_delim_clip, 17
read_delim_dumb, 17
read_tsv_dumb (read_delim_dumb), 17
readr::format_delim(), 21, 22
readr::read_delim(), 17
readr::write_lines(), 22
readr::write_rds(), 22
readr::write_tsv(), 22
save_last(write_last), 22
save_my_packages (load_my_packages), 10
stringr::str_detect(), 9
var_distinct (var_missing), 18
var_missing, 18
view_firefox, 18
view_last, 19
what_in, 5–7, 11–16, 19, 20
what_out, 5–7, 11–16, 20, 20
word_count, 21
write_delim_clip, 21
write_last, 22
24