Package ‘JumpeR’

November 16, 2021

Title Importing and Working with Track and Field Data
Version 0.3.0
Description Primarily used to convert human readable track and field results into dataframes for further analysis. Results can come from central repositories like <https://www.flashresults.com/> or <http://www.deltatiming.com/>, or from individual team sites, like those for colleges. Also contains functions useful for working with track and field data.
License MIT + file LICENSE
Encoding UTF-8
RoxygenNote 7.1.1
Imports magrittr, dplyr, purrr, pdftools, rvest, stringr, SwimmeR, xml2
Suggests testthat
NeedsCompilation no
Author Greg Pilgrim [aut, cre] (<https://orcid.org/0000-0001-7831-442X>), George Perry [ctb]
Maintainer Greg Pilgrim <gpilgrim2670@gmail.com>
Depends R (>= 3.5.0)
Repository CRAN
Date/Publication 2021-11-16 19:40:02 UTC

R topics documented:

  add_row_numbers .................................................. 3
  attempts_remove .................................................... 3
  attempts_split ...................................................... 4
  attempts_split_cols ................................................. 4
  attempts_split_long ............................................... 5
  collect_relay_athletes ............................................ 5
  event_parse .......................................................... 6
  fill_down ............................................................. 7
fill_left ................................................................. 7
flash_attempts_split_long_helper ................................ 8
flash_clean_distance_events ....................................... 8
flash_clean_events .................................................. 9
flash_clean_events_helper ......................................... 10
flash_clean_horizontal_events .................................... 10
flash_clean_relay_events .......................................... 11
flash_clean_sprint_events ......................................... 12
flash_clean_vertical_events ....................................... 12
flash_col_names ..................................................... 13
flash_col_names_helper ............................................ 13
flash_correct_column_overshoot .................................. 14
flash_correct_column_overshoot_helper ........................ 15
flash_date_parse ..................................................... 15
flash_event_links .................................................... 16
flash_event_parse .................................................... 16
flash_extract_details_links ....................................... 17
flash_extract_details_links_helper .............................. 18
flash_gender_parse .................................................. 18
flash_parse .......................................................... 19
flash_parse_table .................................................... 20
flash_pivot_longer ................................................... 20
flash_rebuild_event_table ......................................... 21
flash_rounds_parse ................................................... 21
flash_round_attempts_parse ....................................... 22
flash_year_links ...................................................... 22
hytek_attempts_split_long_helper ................................ 23
hytek_parse .......................................................... 24
is_link_broken ........................................................ 24
lines_sort ............................................................. 25
list_transform ......................................................... 25
math_format ........................................................... 26
math_format_helper ................................................... 27
metric_conversion ..................................................... 28
metric_conversion_helper .......................................... 28
read_results .......................................................... 29
remove_duplicate_splits .......................................... 29
remove_unneeded_rounds .......................................... 30
rounds_parse ........................................................ 30
round_attempts_parse .............................................. 31
splits_parse .......................................................... 31
standard_conversion ............................................... 32
standard_conversion_helper ...................................... 32
tf_parse ............................................................... 33
wind_from_rounds ..................................................... 33
wind_from_rounds_helper .......................................... 35
wind_parse_hytek ...................................................... 35

Index 37
**add_row_numbers**

Add row numbers to raw results

**Description**

Takes the output of `read_results` and adds row numbers to it.

**Usage**

```r
add_row_numbers(text)
```

**Arguments**

- `text`: output from `read_results`.

**Value**

Returns a data frame with event names and row numbers to eventually be recombined with T&F results inside `swim_parse`.

**See Also**

- `add_row_numbers` is a helper function inside `tf_parse`.

---

**attempts_remove**

Collects flight attempts within `tf_parse`.

**Description**

Takes the output of `read_results` and, inside of `tf_parse`, extracts vertical jump attempts and associated row numbers.

**Usage**

```r
attempts_remove(df)
```

**Arguments**

- `df`: dataframe with jump attempt columns containing (X, O, PASS etc) and other columns.

**Value**

Returns a dataframe with the attempt columns removed.

**See Also**

- `attempts_remove` runs inside `flash_parse`.
attempts_split

Description

Given a data frame with columns "Round_1_Attempts" it will output three columns, for each of the attempts in round 1 (Round_1_Attempt_1, Round_1_Attempt_2 etc.)

Usage

attempts_split(data_to_split)

Arguments

data_to_split output from read_results followed by add_row_numbers

Value

returns a data frame with Round_X_Attempts columns split into individual attempts inside tf_parse

See Also

attempts_split is a helper function inside tf_parse

attempts_split_cols

Creates new columns for splitting attempts strings

Description

Given a data frame with columns "Round_1_Attempts" it will produce three columns, for each of the attempts in round 1 (Round_1_Attempt_1, Round_1_Attempt_2 etc.)

Usage

attempts_split_cols(i, data, new_cols, old_cols)

Arguments

i iterative value
data output from tf_parse
new_cols a list of new column names to make
old_cols a list of old columns to split

Value

returns a data frame with Round_X_Attempts columns split into individual attempts inside tf_parse
attempts_split_long

Description

Given a data frame with columns "Round_1_Attempts" it will create three new rows, one for each of the attempts in round 1

Usage

attempts_split_long(data_to_split)

Arguments

data_to_split output from read_results followed by add_row_numbers

Value

returns a data frame with Round_X_Attempts columns split into individual attempts as rows

Examples

df %>% attempts_split_long()

collect_relay_athletes

Description

Collects relay athletes as a data frame within tf_parse

Usage

collect_relay_athletes(x)
Arguments

x          output from read_results followed by add_row_numbers

Value

returns a data frame of relay athletes and the associated performance row number

See Also

collect_relay_athletes_data runs inside of tf_parse

---

**event_parse**

Pulls out event labels from text

**Description**

Locates event labels in text of results output from read_results and their associated row numbers. The resulting data frame is joined back into results to include event names.

**Usage**

event_parse(text)

**Arguments**

text          output from read_results followed by add_row_numbers

**Value**

returns a data frame with event names and row numbers to eventually be recombined with track and field results inside tf_parse

**See Also**

event_parse is a helper function inside tf_parse
fill_down  \hspace{1cm} \textit{Fills NA values with previous non-NA value}

\underline{Description}

This is a base approximation of \texttt{tidyr::fill()}

\underline{Usage}

\texttt{fill_down(x)}

\underline{Arguments}

\begin{itemize}
  \item \textit{x} \hspace{1cm} a list having some number of non-NA values
\end{itemize}

\underline{Value}

a list where NA values have been replaced with the closest previous non-NA value

\underline{See Also}

\begin{itemize}
  \item \texttt{fill_down} is a helper function inside \texttt{lines_sort}
\end{itemize}

fill_left \hspace{1cm} \textit{Shifts non-NA values to left in data frame}

\underline{Description}

Moves non-NA data left into NA spaces, then removes all columns that contain only NA values

\underline{Usage}

\texttt{fill_left(df)}

\underline{Arguments}

\begin{itemize}
  \item \textit{df} \hspace{1cm} a data frame having some NA values
\end{itemize}

\underline{Value}

a data frame where all values have been pushed left, replacing NAs, and all columns containing only NA values have been removed

\underline{See Also}

\begin{itemize}
  \item \texttt{fill_left} is a helper function inside \texttt{lines_sort}
\end{itemize}
**flash_attempts_split_long_helper**

*Creates new columns for splitting attempts strings in long format*

**Description**

Given a data frame with columns "Round_1_Attempts" it will produce three rows, for each of the attempts in flight 1.

**Usage**

`flash_attempts_split_long_helper(data)`

**Arguments**

- `data`: output from `tf_parse`

**Value**

returns a data frame with Round_X_Attempts columns split into individual rows

**See Also**

- `attempts_split_long_helper` is a helper function inside `attempts_split_long`

---

**flash_clean_distance_events**

*Cleans distance events*

**Description**

Cleans distance event results pulled from Flash Results html tables. Distance events are generally those with lengths of 400m or greater. Can present cleaned data in wide or long format.

**Usage**

`flash_clean_distance_events(df, wide_format_distance = wide_format_clean)`

`distance_events(df, wide_format_distance = wide_format_clean)`

**Arguments**

- `df`: a data frame of distance event data from Flash Results
- `wide_format_distance`: should `df` be presented in wide format (default is FALSE)?
**flash_clean_events**

**Value**

a cleaned version of df

**See Also**

flash_clean_distance_events is a helper function inside flash_parse_table

---

**flash_clean_events**  
*Cleans event data*

**Description**

Cleans event results pulled from Flash Results html tables. Can present cleaned data in wide or long format.

**Usage**

```r
flash_clean_events(df, wide_format_clean = FALSE)
clean_results(df, wide_format_clean = FALSE)
```

**Arguments**

- `df`  
a data frame or list of data frames containing event data from Flash Results

- `wide_format_clean`  
should df be presented in wide format (default is FALSE)?

**Value**

a cleaned version of df

**See Also**

flash_clean_events is a helper function inside flash_parse_table
flash_clean_events_helper

Applies appropriate event cleaning function

Description

Used to apply appropriate cleaning function based on event name

Usage

flash_clean_events_helper(
    df_helper = df,
    wide_format_clean_helper = wide_format_clean
)

Arguments

df_helper a data frame of vertical event data from Flash Results
wide_format_clean_helper should df be presented in wide format (default is FALSE)?

Value

a cleaned version of df

See Also

flash_clean_events_helper is a helper function inside flash_clean_events

flash_clean_horizontal_events

Cleans horizontal events

Description

Cleans horizontal event results pulled from Flash Results html tables. Can present cleaned data in wide or long format.

Usage

flash_clean_horizontal_events(df, wide_format_horizontal = wide_format_clean)

horizontal_events(df, wide_format_horizontal = wide_format_clean)
**flash_clean_relay_events**

**Arguments**

- `df` 
  a data frame of horizontal event data from Flash Results
- `wide_format_horizontal` 
  should `df` be presented in wide format (default is FALSE)?

**Value**

- a cleaned version of `df`

**See Also**

- `flash_clean_horizontal_events` is a helper function inside `flash_parse_table`

---

**Description**

Cleans results pulled from Flash Results html tables for relay events. Can present cleaned data in wide or long format.

**Usage**

- `flash_clean_relay_events(df, wide_format_relay)`
- `relay_events(df, wide_format_relay)`

**Arguments**

- `df` 
  a data frame of relay event data from Flash Results
- `wide_format_relay` 
  should `df` be presented in wide format (default is FALSE)?

**Value**

- a cleaned version of `df`

**See Also**

- `flash_clean_relay_events` is a helper function inside `flash_parse_table`
**flash_clean_sprint_events**

*Cleans sprint events*

**Description**

Cleans results pulled from Flash Results html tables for sprint events. Sprint events are generally those with lengths of less than 400m. Can present cleaned data in wide or long format.

**Usage**

```r
flash_clean_sprint_events(df, wide_format_sprint)
sprint_events(df, wide_format_sprint)
```

**Arguments**

- `df` : a data frame of sprint event data from Flash Results
- `wide_format_sprint` : should `df` be presented in wide format (default is `FALSE`)?

**Value**

a cleaned version of `df`

**See Also**

`flash_clean_sprint_events` is a helper function inside `flash_parse_table`

---

**flash_clean_vertical_events**

*Cleans vertical events*

**Description**

Cleans vertical event results pulled from Flash Results html tables. Can present cleaned data in wide or long format.

**Usage**

```r
flash_clean_vertical_events(df, wide_format_vertical = wide_format_clean)
vertical_events(df, wide_format_vertical = wide_format_clean)
```

**Arguments**

- `df` : a data frame of vertical event data from Flash Results
- `wide_format_vertical` : should `df` be presented in wide format (default is `FALSE`)?

**Value**

a cleaned version of `df`

**See Also**

`flash_clean_vertical_events` is a helper function inside `flash_parse_table`
**flash_col_names**

**Arguments**

- df: a data frame of vertical event data from Flash Results
- wide_format_vertical: should df be presented in wide format (default is FALSE)?

**Value**

- a cleaned version of df

**See Also**

- `flash_clean_vertical_events` is a helper function inside `flash_parse_table`

---

`flash_col_names`  
Regularizes column names from Flash Results

**Description**

Split columns have many different naming conventions within Flash Results. This function attempts to enforce one convention, "Split_XXX" where XXX are digits representing distance in meters

**Usage**

`flash_col_names(df)`

**Arguments**

- df: a data frame or list of data frames containing event data from Flash Results

**Value**

- a version of df with split column names renamed

---

`flash_col_names_helper`

*Helper Function for regularizing column names from Flash Results*

**Description**

Helper Function for regularizing column names from Flash Results

**Usage**

`flash_col_names_helper(old_names)`
Arguments

old_names    a list of column names to be reformatted

Value

a list of strings containing corrected split column names

---

`flash_correct_column_overshoot`

**Corrects column index overshoots when naming columns based on their contents**

---

Description

When naming columns based on the contents of a data frame the position of a particular term, e.g. "Athlete" is used to name a column "Athlete". If there is a blank row at the top of the data frame then the position of "Athlete" will be offset by the number of columns in the data frame. This function corrects for that.

Usage

`flash_correct_column_overshoot(x = NA, df)`

Arguments

  x    a column position index

  df    a data frame with missing column names

Value

a correct index for column x

See Also

`flash_correct_column_overshoot` is a helper function inside `flash_parse_table`
Description

When naming columns based on the contents of a data frame the position of a particular term, eg "Athlete" is used to name a column "Athlete". If there is a blank row at the top of the data frame then the position of "Athlete" will be offset by the number of columns in the data frame. This function corrects for that.

Usage

```
flash_correct_column_overshoot_helper(x, df)
```

Arguments

- `x`: a column position index
- `df`: a data frame with missing column names

Value

a correct index for column `x`

See Also

`flash_correct_column_overshoot` is a helper function inside `flash_parse_table`.

Description

Locates an date in text of results from a flash results html page for a given event.

Usage

```
flash_date_parse(text)
```

Arguments

- `text`: raw text of an event page from Flash Results

Value

a one element list containing the date of the event
See Also
flash_date_parse is a helper function inside flash_parse_table

---

**flash_event_links**

*Collects all event result links from a meet landing page on flashresults.com*

---

**Description**

Used in scraping flashresults.com. Collects event result links from a meet landing page

**Usage**

```python
flash_event_links(meet_home)
meet_links(meet_home)
```

**Arguments**

- **meet_home**

  a link to a meet landing page on flashresults.com

**Value**

returns a list of links to individual events from a given meet

**Author(s)**

Gregory A. Pilgrim <gpilgrim2670@gmail.com> and George M. Perry

**Examples**

```python
```

---

**flash_event_parse**

*Pulls out event label from text of flash results html page*

---

**Description**

Locates an event label in text of results from a flash results html page for a given event.

**Usage**

```python
flash_event_parse(text)
```
**flash_extract_details_links**

Collects links to all detailed results links from a given event link on Flash Results

**Description**

Used in scraping flashresults.com. Collects detailed results (often called heat or flight results) from an associated event results landing page. Detailed results often contain splits or attempts results.

**Usage**

```python
flash_extract_details_links(link)
extract_details_links(link)
```

**Arguments**

- **link**
  a link to an event landing page on flashresults.com

**Value**

returns list of links to corresponding detailed event result pages

**Examples**

```python
flash_extract_details_links("https://flashresults.com/2015_Meets/Outdoor/06-25_USATF/008-3Compiled.htm")
```
**flash_extract_details_links_helper**

*Collects links to all detailed results links from a given event link on Flash Results*

**Description**

Used in scraping flashresults.com. Collects detailed results (often called heat or flight results) from an associated event results landing page. Detailed results often contain splits or attempts results.

**Usage**

```r
collect_links(link_helper = link)
```

**Arguments**

- `link_helper`: a link to an event landing page on flashresults.com

**Value**

returns list of links to corresponding detailed event result pages

**See Also**

- `flash_extract_details_links_helper` is a helper function inside `flash_extract_details_links`

---

**flash_gender_parse**

*Pulls out gender label from text of flash results html page*

**Description**

Locates a gender label in text of results from a flash results html page for a given event.

**Usage**

```r
gender_label(text)
```

**Arguments**

- `text`: raw text of an event page from Flash Results

**Value**

- a one element list containing the gender of the event

**See Also**

- `flash_gender_parse` is a helper function inside `flash_parse_table`
flash_parse

Reads track and field results into a list of strings in preparation for parsing with tf_parse

Description

Outputs list of strings to be processed by tf_parse

Usage

```r
flash_parse(
  flash_file,
  flash_rounds = rounds,
  flash_round_attempts = round_attempts,
  flash_split_attempts = split_attempts
)
```

Arguments

- `flash_file` a .pdf or .html file (could be a url) where containing track and field results. Must be formatted in a "normal" fashion - see vignette
- `flash_rounds` should tf_parse try to include rounds for jumping/throwing events? Defaults to FALSE
- `flash_round_attempts` should tf_parse try to include outcomes for rounds for vertical jumping events? Defaults to FALSE
- `flash_split_attempts` should round_attempts columns be split into individual attempts

Value

a data frame of track and field results

See Also

tf_parse is meant to be preceded by read_results
flash_parse_table

Description

Used in scraping flashresults.com. Collects results given in html tables on a specified page into a data frame.

Usage

flash_parse_table(link, wide_format = FALSE, clean = FALSE)
get_results_table(link, wide_format = FALSE, clean = FALSE)

Arguments

link a link to an event landing page on flashresults.com
wide_format should results be presented in wide format (defaults to FALSE)
clean should results be cleaned by flash_clean_events? Default is FALSE.

Value

returns a data frame of results scraped from link

Examples


flash_pivot_longer

Description

Converts Flash Results from wide to long format

Used to convert multiple split columns to two columns, Split_Time and Split_Distance. Effectively a T&F specific version of tidyr::pivot_longer or base::reshape

Usage

flash_pivot_longer(df, varying)

Arguments

df a data frame or list of data frames containing event data from Flash Results
varying names of columns containing varying information (i.e. splits)
**flash_rebuild_event_table**

*Value*

a version of df with split column values as Split_Time and split column names as Split_Distance

---

**flash_rebuild_event_table**

Rebuilds tables that rvest::html_table can’t parse inside of flash_parse_table

---

**Description**

Extracts individual td and th elements from html tables on Flash Results that cannot be parsed by codervest::html_table (due to formatting issues in the html code)

**Usage**

flash_rebuild_event_table(event_url_rebuild)

**Arguments**

event_url_rebuild

a link to an event page on flashresults.com

**Value**

returns a data frame of event results

**See Also**

rebuild_event_table is a helper function inside flash_parse_table

---

**flash_rounds_parse**

Collects attempts within tf_parse

---

**Description**

Takes the output of read_results and, inside of tf_parse, extracts jump/throw attempts and associated row numbers

**Usage**

flash_rounds_parse(text)

**Arguments**

text

output of read_results with row numbers appended by add_row_numbers
Value

returns a data frame with split times and row numbers

See Also

rounds_parse_flash runs inside flash_parse on the output of read_results with row numbers from add_row_numbers
**flash_year_links**

*Collects all meet links from a given year on Flash Results*

**Description**

Used in scraping flashresults.com. Collects meet names, dates, and locations along with a link to the associated results landing page.

**Usage**

```r
flash_year_links(flash_year)
year_links(flash_year)
```

**Arguments**

- `flash_year`: a link to a year landing page on flashresults.com

**Value**

returns a data frame with meet names, dates, locations, and links to flash results

**Examples**

```r
flash_year_links("https://flashresults.com/2015results.htm")
```

---

**hytek_attempts_split_long_helper**

*Creates new columns for splitting attempts strings in long format*

**Description**

Given a data frame with columns "Round_1_Attempts" it will produce three rows, for each of the attempts in flight 1

**Usage**

```r
hytek_attempts_split_long_helper(i, data, old_cols)
```

**Arguments**

- `i`: output from `read_results` followed by `add_row_numbers`
- `data`: output from `tf_parse`
- `old_cols`: a list of old columns to split
Value

returns a data frame with Round_X_Attempts columns split into individual rows

See Also

attempts_split_long_helper is a helper function inside attempts_split_long

hytek_parse  Parses Hytek format track and field results inside tf_parse

Description

Outputs a data frame of track and field results

Usage

```r
hytek_parse(
  hytek_file = file,
  hytek_relay_athletes = relay_athletes,
  hytek_rounds = rounds,
  hytek_round_attempts = round_attempts,
  hytek_split_attempts = split_attempts,
  hytek_splits = splits,
  hytek_split_length = split_length
)
```

Arguments

- **hytek_file** data with row numbers added
- **hytek_relay_athletes** should tf_parse try to include the names of relay athletes for relay events? Names will be listed in new columns "Relay-Athlete_1", "Relay_Athlete_2" etc. Defaults to FALSE.
- **hytek_rounds** should tf_parse try to include rounds for jumping/throwing events? Please note this will add a significant number of columns to the resulting data frame. Defaults to FALSE.
- **hytek_round_attempts** should tf_parse try to include rounds results (i.e. "PASS", "X", "O") for high jump and pole value events? Please note this will add a significant number of columns to the resulting data frame. Defaults to FALSE.
- **hytek_split_attempts** should tf_parse split attempts from each round into separate columns? For example "XXO" would result in three columns, one for "X", another for the second "X" and third for "O". There will be a lot of columns. Defaults to FALSE.
- **hytek_splits** either TRUE or the default, FALSE - should hytek_parse attempt to include splits.
is_link_broken

hytek_split_length

either the distance at which splits are collected (must be constant distance) or
the default, 1, the length of track at which splits are recorded. Not all results are
internally consistent on this issue. If in doubt use the default 1

Value

a data frame of track and field results

See Also

hytek_parse is runs inside tf_parse

is_link_broken

Determines if a link is valid

Description

Used in testing links to external data, specifically inside of internal package tests. Attempts to
connect to link for the length of duration (in s). If it fails it returns TRUE

Usage

is_link_broken(link_to_test, duration = 1)

Arguments

link_to_test a link
duration the lowest row number

Value

FALSE if the link works, TRUE if it fails

lines_sort

Sorts and collects lines by performance and row number

Description

Collects all lines, (for example containing splits or relay swimmers) associated with a particular
performance into a data frame with the appropriate row number for that performance.

Usage

lines_sort(x, min_row = minimum_row)
Arguments

- **x**: a list of character strings including performances, with row numbers added by `add_row_numbers`
- **min_row**: the lowest row number

Value

A data frame with `Row_Numb` as the first column. Other columns are performance elements, like splits or relay swimmers, both in order of occurrence left to right.

---

**list_transform**  
Transform list of lists into data frame

---

Description

Converts list of lists, with all sub-lists having the same number of elements into a data frame where each sub-list is a row and each element a column.

Usage

```r
list_transform(x)
```

Arguments

- **x**: a list of lists, with all sub-lists having the same length

Value

A data frame where each sub-list is a row and each element of that sub-list is a column.

See Also

- `list_transform` is a helper function used inside of `tf_parse` and `event_parse`
**Description**

Takes a character string (or list) representing time in track format (e.g. 1:35.37) and converts it to a numeric value (95.37) or a list of values representing seconds.

**Usage**

`math_format(x)`

**Arguments**

- `x` A character vector of time(s) in track format (e.g. 1:35.93, as minutes:seconds.tenths hundredths) to be converted to seconds (95.93)

**Value**

returns the value of the string `x` which represents a time in track format (mm:ss.th) and converts it to seconds

**Examples**

```r
math_format("1:35.93")
math_format("16:45.19")
math_format("25.43")
math_format(c("1:35.93", "16:45.19", NA, "25.43"))
```

**math_format_helper**  
*Helper function for formatting mm:ss.th times as seconds*

**Description**

Helper function for formatting mm:ss.th times as seconds

**Usage**

`math_format_helper(x)`

**Arguments**

- `x` A character vector of time(s) in track format (e.g. 1:35.93) to be converted to seconds (95.93)
Value

a numeric value representing a time or distance. Units are not included

---

**metric_conversion**

*Formatting feet-inches lengths as meters*

*Description*

Takes a character string (or list) representing a length in feet-inches format (e.g. "12-07.45") and converts it to a distance in meters ("3.85m").

*Usage*

```r
metric_conversion(x)
```

*Arguments*

- `x`: A character vector of distance(s) in feet-inches format (e.g. "12-07.45"), to be converted to meters ("3.85m")

*Value*

returns the value of the string `x` which represents a distance in meters, as a character, with unit "m" included

*Examples*

```r
distances <- c("1.23m", "5-02.34", "43.45", "6.89", NA)
m <- metric_conversion(distances)
math_format(metric_conversion(distances))
m <- metric_conversion("5.45m")
```

---

**metric_conversion_helper**

*Converts distances in feet-inches to meters*

*Description*

Converts distances in feet-inches to meters

*Usage*

```r
metric_conversion_helper(x)
```

*Arguments*

- `x`: A character vector of distance(s) to be converted from feet-inches to meters
**Value**

a numeric value representing a number of meters. Units are not included

---

**read_results**

*Reads track and field results into a list of strings in preparation for parsing with tf_parse*

---

**Description**

Outputs list of strings to be processed by tf_parse

**Usage**

```r
read_results(file, node = "pre")
```

**Arguments**

- `file`: a .pdf or .html file (could be a url) where containing swimming track and field results. pdfs with multiple columns will not work.
- `node`: a CSS node where html results are stored. Required for html results. Default is "pre", which nearly always works.

**Value**

returns a list of strings containing the information from `file`. Should then be parsed with tf_parse

**See Also**

`read_results` is meant to be followed by `tf_parse`

**Examples**

```r
read_results("https://www.flashresults.com/2018_Meets/Outdoor/05-05_A10/015-1.pdf")
```
remove_duplicate_splits

*Removes duplicate splits*

**Description**

Removes duplicate splits

**Usage**

```r
remove_duplicate_splits(x)
```

**Arguments**

- `x`: a list of splits, in which position 2 and position 3 might be duplicates

**Value**

a list with duplicated value in position 2 removed

**See Also**

`remove_duplicate_splits` is a helper function inside `splits_parse`

---

remove_unneeded_rounds

*Removes unneeded rounds columns within tf_parse*

**Description**

Inside of `tf_parse` & `tf_parse`, removes round columns that do not have an associated `round_attempts` column

**Usage**

```r
remove_unneeded_rounds(x)
```

**Arguments**

- `x`: data frame with columns called both "Round_X" and "Round_X_Results" where X is a number

**Value**

returns a data frame where Round_X columns that do not have a corresponding Round_X_Results have been removed
rounds_parse

See Also
remove_unneeded_rounds runs inside flash_parse & tf_parse

---

rounds_parse  Collects rounds within tf_parse

Description
Takes the output of read_results and, inside of tf_parse, extracts jump/throw rounds and associated row numbers.

Usage
rounds_parse(text)

Arguments
text  output of read_results with row numbers appended by add_row_numbers

Value
returns a data frame with split times and row numbers

See Also
rounds_parse runs inside tf_parse on the output of read_results with row numbers from add_row_numbers

---

round_attempts_parse  Collects results of high jump & pole vault attempts within tf_parse

Description
Takes the output of read_results and, inside of tf_parse, extracts jump/throw attempts and associated row numbers.

Usage
round_attempts_parse(text)

Arguments
text  output of read_results with row numbers appended by add_row_numbers

Value
returns a data frame with split times and row numbers


splits_parse

Collects splits within tf_parse

Description

Takes the output of read_results and, inside of tf_parse, extracts split times and associated row numbers

Usage

splits_parse(text, split_len = 1)

Arguments

text

output of read_results with row numbers appended by add_row_numbers

split_len

the distance at which splits are measured

Value

returns a data frame with split times and row numbers

See Also

splits_parse runs inside tf_parse on the output of read_results with row numbers from add_row_numbers

standard_conversion

Formatting meters lengths as feet-inches

Description

Takes a character string (or list) representing a length in meters format (e.g. "3.85m") and converts it to a distance in feet-inches ("12-07.45")

Usage

standard_conversion(x)

Arguments

x

A character vector of distance(s) in meters format ("3.85m") , to be converted to meters ("12-07.45")
**standard_conversion_helper**

Value

returns the value of the string x which represents a distance in feet-inches

---

**tf_parse**

*Parses track and field data from Flash or Hytek format data into a data frame*

**Description**

Outputs a data frame containing track and field data

**Usage**

```r
tf_parse(
  file,
  avoid = avoid_default,
  typo = typo_default,
  replacement = replacement_default,
  relay_athletes = FALSE,
  rounds = FALSE,
  round_attempts = FALSE,
  split_attempts = FALSE,
  splits = FALSE,
  split_length = 1
)
```
### Arguments

- **file**: a .pdf or .html file (could be a url) where containing track and field results. Must be formatted in a "normal" fashion - see vignette
- **avoid**: a list of strings. Rows in file containing these strings will not be included. For example "Record:", often used to label records, could be passed to avoid. The default is avoid_default, which contains many strings similar to "Record:". Users can supply their own lists to avoid.
- **typo**: a list of strings that are typos in the original results. tf_parse is particularly sensitive to accidental double spaces, so "Central High School", with two spaces between "Central" and "High" is a problem, which can be fixed. Pass "Central High School" to typo.
- **replacement**: a list of fixes for the strings in typo. Here one could pass "Central High School" (one space between "Central" and "High") to fix the issue described in typo
- **relay_athletes**: should tf_parse try to include the names of relay athletes for relay events? Names will be listed in new columns "Relay-Athlete_1", "Relay_Athlete_2" etc. Defaults to FALSE.
- **rounds**: should tf_parse try to include rounds for jumping/throwing events? Please note this will add a significant number of columns to the resulting data frame. Defaults to FALSE.
- **round_attempts**: should tf_parse try to include rounds results (i.e. "PASS", "X", "O") for high jump and pole value events? Please note this will add a significant number of columns to the resulting data frame. Defaults to FALSE
- **split_attempts**: should tf_parse split attempts from each round into separate columns? For example "XXO" would result in three columns, one for "X", another for the second "X" and third for "O". There will be a lot of columns. Defaults to FALSE
- **splits**: either TRUE or the default, FALSE - should tf_parse attempt to include splits.
- **split_length**: either the distance at which splits are collected (must be constant distance) or the default, 1, the length of track at which splits are recorded. Not all results are internally consistent on this issue. If in doubt use the default 1

### Value

A data frame of track and field results

### See Also

tf_parse is meant to be preceded by read_results

### Examples

```r
tf_parse(
  read_results("https://www.flashresults.com/2018_Meets/Outdoor/05-05_A10/015-1.pdf"),
  rounds = TRUE,
  round_attempts = TRUE,
  split_attempts = TRUE)
```
wind_from_rounds

Pulls Wind Data by Round from Horizontal Flash Table Results

Description
In some Flash Table results for horizontal events (long jump, triple jump, throwing events), a wind value is listed for each round/attempt. This function pulls out those wind values into columns called "Round_1_Wind" (if the round data is in a column called Round_1)

Usage
wind_from_rounds(df)

Arguments
df a data frame containing results with wind data included in round columns.

Value
a data frame with all wind data in separate (tidy) columns

See Also
wind_from_rounds is a helper function inside flash_clean_horizontal_events

wind_from_rounds_helper

Helper function for extracting wind data from round columns

Description
Helper function for extracting wind data from round columns

Usage
wind_from_rounds_helper(df = df, i, round_cols, ...)

Arguments
df a data frame containing round columns with both results and wind data
i list of values to iterate along
round_cols list of columns containing results and wind values by round
... other arguments as needed

Value
a list of data frames with all wind data for each round in a separate (tidy) column
**wind_parse_hytek**

Collects splits within `tf_parse`

## Description

Takes the output of `read_results` and, inside of `tf_parse`, extracts split times and associated row numbers.

## Usage

```r
wind_parse_hytek(text)
```

## Arguments

- `text`:
  
  output of `read_results` with row numbers appended by `add_row_numbers`

## Value

returns a data frame with wind speeds and row numbers

## See Also

`wind_parse_hytek` runs inside `hytek_parse` on the output of `read_results` with row numbers from `add_row_numbers`
Index

add_row_numbers, 3, 22, 31, 32, 36
attempts_remove, 3
attempts_split, 4
attempts_split_cols, 4
attempts_split_long, 5

clean_results (flash_clean_events), 9
collect_relay_athletes, 5

distance_events
  (flash_clean_distance_events), 8

event_parse, 6
extract_details_links
  (flash_extract_details_links), 17

fill_down, 7
fill_left, 7
flash_attempts_split_long_helper, 8
flash_clean_distance_events, 8
flash_clean_events, 9, 10
flash_clean_events_helper, 10
flash_clean_horizontal_events, 10, 35
flash_clean_relay_events, 11
flash_clean_sprint_events, 12
flash_clean_vertical_events, 12
flash_col_names, 13
flash_col_names_helper, 13
flash_correct_column_overshoot, 14
flash_correct_column_overshoot_helper, 15
flash_date_parse, 15
flash_event_links, 16
flash_event_parse, 16
flash_extract_details_links, 17, 18
flash_extract_details_links_helper, 18
flash_gender_parse, 18
flash_parse, 3, 19, 22, 31

flash_parse_table, 9, 11–18, 20, 21
flash_pivot_longer, 20
flash_rebuild_event_table, 21
flash_round_attempts_parse, 22
flash_rounds_parse, 21
flash_year_links, 23
get_results_table (flash_parse_table), 20

horizontal_events
  (flash_clean_horizontal_events), 10
hytek_attempts_split_long_helper, 23
hytek_parse, 24, 36
is_link_broken, 25

lines_sort, 25
list_transform, 26

math_format, 27
math_format_helper, 27
meet_links (flash_event_links), 16
metric_conversion, 28
metric_conversion_helper, 28

read_results, 19, 22, 29, 31, 32, 34, 36
relay_events
  (flash_clean_relay_events), 11
remove_duplicate_splits, 30
remove_unneeded_rounds, 30
round_attempts_parse, 31
rounds_parse, 31

splits_parse, 32
sprint_events
  (flash_clean_sprint_events), 12
standard_conversion, 32
standard_conversion_helper, 33
tf_parse, 3, 4, 6, 25, 29, 31, 32, 33

vertical_events
   (flash_clean_vertical_events), 12

wind_from_rounds, 35
wind_from_rounds_helper, 35
wind_parse_hytek, 36

year_links (flash_year_links), 23