Package: peranavolley (via r-universe)

September 27, 2024

Title Perana Sports Volleyball Files

Version 0.7.6

Description Basic functions for reading and working with Perana Sports volleyball scouting files.

Depends R (>= 3.3.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports assertthat, base64enc, datavolley, dplyr, lubridate, jsonlite, rlang, stringi, stringr, tidyr

RoxygenNote 7.1.2

Suggests knitr, testthat

Remotes openvolley/datavolley

VignetteBuilder knitr

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

RemoteUrl https://github.com/openvolley/peranavolley

RemoteRef HEAD

RemoteSha 1198807ad01791003e5543263c7a92160da7accf

Contents

`peranavolley` **peranavolley**

Description

Functions for reading and working with Perana Sports volleyball scouting files.

Author(s)

Ben Raymond <ben@untan.gl>

`plays` *Extract the plays component from a peranavolley object*

Description

Extract the plays component from a peranavolley object

Usage

```
plays(x)
```

Arguments

`x`	peranavolley: a peranavolley object as returned by `pv_read`

Value

The plays component of x (a data.frame)

See Also

`pv_read`

Examples

```
## Not run:
  x <- pv_read(pv_example_file())
  inspect(plays(x))

## End(Not run)
```

`print.summary.peranavolley` *Print method for summary.peranavolley*

Description

Print method for summary.peranavolley

Usage

```
## S3 method for class 'summary.peranavolley'
print(x, ...)
```

Arguments

`x`	summary.peranavolley: a summary.peranavolley object as returned by `summary.peranavolley`
`...`	: additional arguments (currently these have no effect)

See Also

`summary.peranavolley`

`pt_example_file` *Example psvt data files provided as part of the peranavolley package*

Description

Example psvt data files provided as part of the peranavolley package

Usage

```
pt_example_file(choice = 1)
```

Arguments

`choice`	numeric: which data file to return? • 1 - trivial volleyball example file

Value

path to the file

See Also

pt_read

Examples

```
myfile <- pt_example_file()
x <- pt_read(myfile)
```

---

pt_read *Read a Perana Sports tagger (psvt) data file*

Description

Read a Perana Sports tagger (psvt) data file

Usage

```
pt_read(filename, raw_only = FALSE)
```

Arguments

filename	string: path to file
raw_only	logical: if TRUE, just decompress the file, don't parse it

Value

A list with elements meta (metadata including the project and template information, and information about the skills and grades used in the tags) and tags the tag data.

References

http://peranasports.com/

pt_write *Write a Perana Sports tagger (psvt) data file*

Description

This is somewhat experimental. It may be useful if one wants to read an existing file, modify the content, and re-write it back to a psvt file.

Usage

```
pt_write(x, filename)
```

Arguments

x	character: data to write. See e.g the raw component of the object returned by pt_read
filename	string: path to file

See Also

pt_read

pv_default_errortypes *Define errortype interpretations*

Description

Define the interpretation of errortypes associated with events.

Usage

```
pv_default_errortypes()
```

Value

a tibble with columns "skill", "errortype", and "evaluation"

See Also

pv_read

Examples

```
pv_default_errortypes()
```

pv_default_eventgrades *Define eventgrade interpretations*

Description

Define the interpretation of eventgrades associated with events.

Usage

```
pv_default_eventgrades()
```

Value

a tibble with columns "skill", "eventgrade", "evaluation_code" (the equivalent DataVolley code, if there is one), "evaluation", and "win_loss"

See Also

pv_read

Examples

```
pv_default_eventgrades()
```

pv_default_subevents *Define subevent interpretations*

Description

Define the interpretation of subevents associated with events.

Usage

```
pv_default_subevents()
```

Value

a tibble with columns "skill", "subevent", and "evaluation"

See Also

pv_read

Examples

```
pv_default_subevents()
```

pv_example_file *Example data files provided as part of the peranavolley package*

Description

These example files were kindly provided by Chau Le from Perana Sports.

Usage

```
pv_example_file(choice = 1)
```

Arguments

choice	numeric: which data file to return? • 1 - Men's Australian Volleyball League 2017: Canberra Heat vs UTSSU • 2 - Women's Australian Volleyball League 2017: Canberra Heat vs UTSSU

Value

path to the file

See Also

pv_read

Examples

```
myfile <- pv_example_file()
x <- pv_read(myfile)
summary(x)
```

pv_read *Read Perana Sports volleyball data file*

Description

Read Perana Sports volleyball data file

Usage

```
pv_read(
  filename,
  insert_technical_timeouts = FALSE,
  do_warn = FALSE,
  extra_validation = 2,
  raw_only = FALSE,
  eventgrades = pv_default_eventgrades(),
  errortypes = pv_default_errortypes(),
  subevents = pv_default_subevents(),
  setting_zones,
  postprocess = NULL
)
```

Arguments

filename	string: path to file
insert_technical_timeouts	logical: if TRUE, insert technical timeouts at 8 and 16 points
do_warn	logical: should we issue warnings about the contents of the file as we read it?
extra_validation	numeric: should we run some extra validation checks on the file? 0=no extra validation, 1=check only for major errors, 2=somewhat more extensive, 3=the most extra checking
raw_only	logical: if TRUE, just decompress the file, don't parse it
eventgrades	tibble: a tibble that defines the interpretations of eventgrade values; see pv_default_eventgrades
errortypes	tibble: a tibble that defines the interpretations of errortype values; see pv_default_errortypes
subevents	tibble: a tibble that defines the interpretations of subevent values; see pv_default_subevents
setting_zones	named character: if the data file has been scouted using setting zones, then each attack will have its associated setting zone (numbered 1 to 5). The setting zone names are not stored in the file, so they can be provided here as a character vector. This can either be an un-named character vector, in which case it must be of length 5; otherwise if only a subset of the five setting zones are being used then it can be provided as a named character vector(e.g. setting_zones = c("1" = "X1", "3" = "X7", "4" = "medium/fast", "5" = "high")). The values in this vector will be used to populate the attack_code column of the returned plays data
postprocess	string or function: function, or name of function, to apply to the peranavolley object as the final step in the processing

Value

A named list with several elements. raw contains the extracted but unparsed text from the psvb file, meta provides match metadata, plays the play-by-play data in the form of a data.frame, and messages is a data.frame describing any inconsistencies found in the file.

References

http://peranasports.com/

See Also

pv_validate, pv_default_eventgrades, pv_default_errortypes

Examples

```
filename <- pv_example_file()
x <- pv_read(filename)

x <- pv_read(filename, setting_zones = c("X1", "X2", "X7", "medium/fast", "high"))
```

---

pv_recode *Generate more meaningful entries in plays data*

Description

Notes:

- the rules are applied in order, so that a row in x that matches multiple rows in remap$conditions will have the first matching row applied to it. Similarly, if remap is provided as a list of remaps, then these will be applied in order and only the first matching one will be applied to any given row

Usage

```
pv_recode(x, remap, log_changes = FALSE)
```

Arguments

x	data.frame or tibble: a peranavolley object as returned by pv_read, or the plays component thereof
remap	list: a remapping is defined by a list with two data.frames. The data.frame named "conditions" defines the conditions to match on, and the one named "values" provides the new values to use. See the example provided and link{pv_tas_remap} for another. The remap parameter to pv_recode can be provided as one such remapping, or it can be provided as a list of remappings (i.e. a list of lists). In this case the list should be named, and those names will be used in the "changes" attribute (see log_changes)
log_changes	logical: if TRUE, the returned object will have a "changes" attribute describing the changes that were made

Value

A copy of x with new values applied

Examples

```
## read file
x <- pv_read(pv_example_file())

## construct the remapping
my_remap <- list(conditions =
                   data.frame(attack_code = c("1", NA_character_, "4", "5"),
                              start_zone = c(NA, 3, 2, 2),
                              stringsAsFactors = FALSE),
                 values =
                   data.frame(attack_code = c("X1", "Z3", "X6", "V6"),
                              stringsAsFactors = FALSE))

## meaning that: any attack with attack_code 1 will be recoded as an "X1" attack
##               any attack from start_zone 3 will be recoded as an "Z3" attack
##          any attack from start_zone 2 with attack_code 4 will be recoded as an "X6" attack
##          any attack from start_zone 2 with attack_code 5 will be recoded as an "V6" attack

x <- pv_recode(x, remap = my_remap)
```

`pv_tas_data_augment` *Augment partially-scouted data*

Description

Augment partially-scouted data

Usage

```
pv_tas_data_augment(x)
```

Arguments

`x`	peranavolley: a peranavolley object as returned by `pv_read`

Value

A peranavolley object with additional rows added to its `plays` component.

See Also

`pv_read`, `pv_tas_recode`, `pv_tas_live_recode`

pv_tas_recode *An opinionated recoding*

Description

Note that this function is only likely to be useful if you have scouted your VBStats files following the conventions described at https://raymondben.github.io/scouting-notes/

Usage

```
pv_tas_recode(x, remap = pv_tas_remap(), log_changes = FALSE)

pv_tas_live_recode(
  x,
  remap = pv_tas_remap(),
  home_team_rotation = NULL,
  visiting_team_rotation = NULL,
  log_changes = FALSE
)

pv_tas_recode_augment(x, remap = pv_tas_remap(), log_changes = FALSE)

pv_tas_live_recode_augment(
  x,
  remap = pv_tas_remap(),
  home_team_rotation = NULL,
  visiting_team_rotation = NULL,
  log_changes = FALSE
)

pv_tas_remap()
```

Arguments

x	data.frame or tibble: a peranavolley object as returned by pv_read, or the plays component thereof
remap	list: a list with components "conditions" and "values" that define the remapping. See pv_tas_remap for an example
log_changes	logical: if TRUE, the returned object will have a "changes" attribute describing the changes that were made
home_team_rotation	string: fixed rotation of the home team, either "SHM" or "SMH". This will be used to infer missing attack start_zone values (assuming the standard attack by role - e.g. the default attack for an opposite is X6 when they are front row (except in P1 reception when it is X5), and X8 when they are back row). If home_team_rotation is NULL this will not be done

visiting_team_rotation
: string: as for home_team_rotation, but for the visiting team

Details

This function modifies the plays component of a peranavolley object, following the scouting conventions described at https://raymondben.github.io/scouting-notes/. The following changes are made:

- attacks marked as "Setter tip" ("Dump" in VBStats) by a setter are mapped to attack_code "PP"
- any "Setter tip" by a non-setter are treated as freeballs. Their skill entry is changed to "Freeball", and attack code to NA
- attack_code 1 to "X1"
- attack_code 2 to "X2"
- attack_code 3 to "X7"
- attack_code 4 to "CF" (slide)
- attack code 5 treated as highball, with new attack code determined by start_zone (V5, V6, etc)
- any other attack (with NA attack code) treated as standard tempo with new attack code determined by start_zone (X5, X6, etc)
- any dig following a freeball over is changed to skill "Freeball"

pv_tas_recode_augment and pv_tas_live_recode_augment are convenience functions that call pv_tas_recode or pv_tas_live_recode followed by pv_tas_data_augment

Value

A modified version of x.

Examples

```
## Note: this is a silly example, because the example VBStats file was NOT
##       scouted with the conventions expected by this function

x <- pv_read(pv_example_file())
x <- pv_tas_recode(x, remap = pv_tas_remap(), log_changes = TRUE)
```

---

pv_validate *Validate parsed Perana Volleyball scouting data*

Description

Validate parsed Perana Volleyball scouting data

Usage

```
pv_validate(x, validation_level = 2)
```

Arguments

x	peranavolley: peranavolley object as returned by pv_read
validation_level	numeric: how strictly to check? If 0, perform no checking; if 1, only identify major errors; if 2, also return any issues that are likely to lead to misinterpretation of data; if 3, return all issues (including minor issues)

Value

data.frame with columns message (the validation message), file_line_number (the corresponding line number in the file), video_time, and file_line (the actual line from the file).

See Also

pv_read

Examples

```
filename <- pv_example_file()
x <- pv_read(filename)
pv_validate(x)
```

---

pv_write *Write a Perana Sports volleyball data file*

Description

This is somewhat experimental. It may be useful if one wants to read an existing file, modify the content, and re-write it back to a PSVB file.

Usage

```
pv_write(x, filename)
```

Arguments

x	character: data to write. See e.g the raw component of the object returned by pv_read
filename	string: path to file

See Also

pv_read

Examples

```
## Not run:
 x <- pv_read(pv_example_file())
 new_file_name <- tempfile(fileext = ".psvb")
 pv_write(x$raw, filename = new_file_name)

## End(Not run)
```

`summary.peranavolley` *A simple summary of a volleyball match*

Description

A simple summary of a volleyball match

Usage

```
## S3 method for class 'peranavolley'
summary(object, ...)
```

Arguments

`object`	peranavolley: peranavolley object as returned by `pv_read`
`...`	: additional arguments (currently these have no effect)

Value

list of summary items

See Also

`pv_read`

Examples

```
x <- pv_read(pv_example_file())
summary(x)
```

Index