Package: tallieR (via r-universe)

July 1, 2026

Title Import and Score ScoreMe App Questionnaire Data

Version 0.2.0

Description A companion package for the ScoreMe app (https://scoreme.circadia-lab.uk). Provides tools to import participant JSON exports, re-score validated questionnaires, and assemble tidy wide and long study-level data frames ready for downstream analysis in R. Stable instruments: ESS, ISI, DBAS-16, MEQ, PSQI, RU-SATED, STOP-BANG, KSS, MCTQ. Beta instruments (scoring ported from ScoreMe, not yet independently validated in tallieR): PHQ-2, PHQ-9, PHQ-15, GAD-7, GAD-2, BDI-II, BAI, DASS-21, PANSS, STAI-S, STAI-T, WHOQOL-BREF, MacArthur SSS, IPAQ-S, GPAQ, GSQ, AQ-10.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Depends R (>= 4.1.0)

Imports jsonlite (>= 1.8.0), dplyr (>= 1.1.0), tidyr (>= 1.3.0), purrr (>= 1.0.0), rlang (>= 1.1.0), cli (>= 3.6.0)

Suggests ggplot2 (>= 3.4.0), knitr, pkgdown, rmarkdown, testthat (>= 3.0.0), tibble (>= 3.0.0)

Config/testthat/edition 3

URL `https://tallier.circadia-lab.uk`, `https://github.com/circadia-bio/tallieR`

BugReports `https://github.com/circadia-bio/tallieR/issues`

Config/pak/sysreqs libicu-dev

Repository https://circadia-bio.r-universe.dev

Date/Publication 2026-05-22 12:30:07 UTC

RemoteUrl https://github.com/circadia-bio/tallieR

RemoteRef main

RemoteSha 7e4878c4015551037d2c8299c82aea6c4e18b502

Contents

`as_tibble.tallier_export` *Coerce a tallier_export to a tibble*

Description

Converts a `tallier_export` object to a tibble by calling `scores_wide()` and coercing the result. One row per participant, one column per questionnaire, with participant metadata prepended by default.

Usage

```
as_tibble.tallier_export(x, ...)
```

Arguments

`x`	A `tallier_export` object.
`...`	Additional arguments passed to `scores_wide()` (e.g. `include_meta = FALSE`).

Details

This method is registered for `tibble::as_tibble()` and is available automatically when the `tibble` package is loaded.

Value

A tibble; see `scores_wide()` for column details.

Examples

```
## Not run:
exp <- read_scoreme("export.json")
tibble::as_tibble(exp)
tibble::as_tibble(exp, include_meta = FALSE)

## End(Not run)
```

`as_tibble.tallier_study`

Coerce a tallier_study to a tibble

Description

Converts a `tallier_study` object to a tibble by calling `scores_wide()` and coercing the result. One row per participant, one column per questionnaire, with participant metadata prepended by default.

Usage

```
as_tibble.tallier_study(x, ...)
```

Arguments

`x`	A `tallier_study` object.
`...`	Additional arguments passed to `scores_wide()` (e.g. `include_meta = FALSE`).

Details

This method is registered for `tibble::as_tibble()` and is available automatically when the `tibble` package is loaded.

Value

A tibble; see `scores_wide()` for column details.

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")
tibble::as_tibble(study)

## End(Not run)
```

`available_instruments` *List available instruments*

Description

Returns a data frame listing all built-in questionnaires supported by tallieR, with their IDs, full titles, clinical domain, maximum score, and beta status.

Usage

```
available_instruments()
```

Details

Beta instruments (`beta = TRUE`) are included in ScoreMe but have not yet been through full validation review in tallieR. Scoring algorithms match the ScoreMe app exactly; use with appropriate caution in clinical contexts.

Value

A `data.frame` with columns `id`, `title`, `domain`, `max_score`, `beta`.

Examples

```
available_instruments()

# View only stable instruments
subset(available_instruments(), !beta)

# View only beta instruments
subset(available_instruments(), beta)
```

`completion_summary` *Completion summary*

Description

Returns a data frame showing which questionnaires each participant has completed. Useful for monitoring data collection progress in longitudinal or multi-site studies.

Usage

```
completion_summary(obj, wide = FALSE, include_date = TRUE, include_meta = TRUE)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`wide`	Logical. If `FALSE` (default), returns a long data frame with one row per participant × questionnaire and a `completed` logical column. If `TRUE`, returns a wide data frame with one row per participant and one logical column per questionnaire.
`include_date`	Logical. If `TRUE` (default), adds a `completed_at` column in long format showing the timestamp of the most recent administration. Ignored when `wide = TRUE`.
`include_meta`	Logical. If `TRUE` (default), participant metadata columns are prepended.

Details

When a participant has completed a questionnaire more than once, it is counted as complete and the most recent `completed_at` timestamp is reported (when `include_date = TRUE`).

Value

In long format: a `data.frame` with columns: participant metadata (optional), `questionnaire_id`, `completed` (logical), and optionally `completed_at` (character timestamp of most recent administration). In wide format: a `data.frame` with one row per participant and one logical column per questionnaire.

See Also

`scores_wide()`, `scores_long()`

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")

# Long format: one row per participant x questionnaire
completion_summary(study)
```

```
# Wide format: one row per participant
completion_summary(study, wide = TRUE)

# Without timestamps
completion_summary(study, include_date = FALSE)

## End(Not run)
```

`cronbach_alpha` *Cronbach's alpha for one or more questionnaires*

Description

Computes Cronbach's alpha (a measure of internal consistency) for each questionnaire present in a `tallier_export` or `tallier_study` object. Item-level responses are extracted via `items_long()`, coerced to numeric, and a participant × item matrix is constructed per questionnaire.

Usage

```
cronbach_alpha(obj, questionnaires = NULL, conf_level = 0.95, min_items = 2L)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object, or a data frame as returned by `items_long()` (must contain columns `participant_id`, `questionnaire_id`, `item_id`, `completed_at`, `response`).
`questionnaires`	Character vector of questionnaire IDs to include. Defaults to all questionnaires present in `obj`.
`conf_level`	Numeric. Confidence level for the CI (default `0.95`). Uses the exact F-distribution method of Feldt et al. (1987).
`min_items`	Integer. Minimum number of numeric items required to attempt estimation (default `2`).

Details

Non-numeric items (e.g. MCTQ clock times, STOP-BANG yes/no) are silently dropped before estimation. Questionnaires with fewer than 2 numeric items or fewer than 2 complete observations return `NA` with an explanatory note.

Value

A `data.frame` with one row per questionnaire and columns:

questionnaire_id Questionnaire identifier.

alpha Cronbach's alpha (standardised scale: $-\infty$ to 1).

ci_lower Lower bound of the `conf_level` CI.

ci_upper Upper bound of the `conf_level` CI.

n_items Number of numeric items used.

n_obs Number of complete observations (participants).

note `NA` on success, or a short message explaining why estimation failed.

References

Cronbach, L. J. (1951). Coefficient alpha and the internal structure of tests. *Psychometrika*, 16(3), 297–334. doi:10.1007/BF02310555

Feldt, L. S., Woodruff, D. J., & Salih, F. A. (1987). Statistical inference for coefficient alpha. *Applied Psychological Measurement*, 11(1), 93–103. doi:10.1177/014662168701100107

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")

# All questionnaires
cronbach_alpha(study)

# Specific subset
cronbach_alpha(study, questionnaires = c("ess", "isi", "phq9"))

# From an items_long() data frame (e.g. already filtered)
items <- items_long(study)
cronbach_alpha(items)

## End(Not run)
```

`interpret_all` *Interpret all questionnaire scores in an export*

Description

Returns a long data frame with one row per participant x questionnaire x administration, augmented with clinical interpretation columns (`label`, `color`, `description`). Mirrors the shape of `scores_long()` so the two can be joined by `participant_id` + `questionnaire_id` + `completed_at`.

Usage

```
interpret_all(obj, include_meta = TRUE, instruments = NULL)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`include_meta`	Logical. If `TRUE` (default), participant metadata columns are prepended (same columns as `scores_long()`).
`instruments`	An optional named list of additional registry entries from `load_instrument()` or `load_instrument_dir()`.

Details

Scores that cannot be interpreted (unknown instrument, `NA` score, or composite score with no matching band) return `NA` in all three interpretation columns rather than an error, so the rest of the study data is unaffected.

Value

A `data.frame` with columns: participant metadata (optional), `questionnaire_id`, `completed_at`, `score`, `label`, `color`, `description`.

See Also

`interpret_score()`, `scores_long()`

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")
interps <- interpret_all(study)

# Join with scores_long() if you need both
scores <- scores_long(study)
full   <- merge(scores, interps[
  c("participant_id", "questionnaire_id", "completed_at",
    "label", "color", "description")
], by = c("participant_id", "questionnaire_id", "completed_at"),
  all.x = TRUE)

## End(Not run)
```

`interpret_score` *Interpret a questionnaire score*

Description

Returns a clinical interpretation for a given score on a given questionnaire, matching the score bands used in the ScoreMe app.

Usage

```
interpret_score(id, score, instruments = NULL)
```

Arguments

`id`	Character. Questionnaire identifier.
`score`	Numeric score (or named list for composite instruments).
`instruments`	An optional named list of additional registry entries from `load_instrument()` or `load_instrument_dir()`.

Value

A list with elements `label`, `color` (hex), and `description`.

Examples

```
interpret_score("ess", 12)
interpret_score("meq", 65)
```

`items_long` *Item-level long table*

Description

Returns a data frame with one row per participant × questionnaire administration × item. Useful for factor analysis, IRT, or item-level reliability checks.

Usage

```
items_long(obj, include_meta = TRUE, scored_items = FALSE, instruments = NULL)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`include_meta`	Logical. If `TRUE` (default), participant metadata columns are included.
`scored_items`	Logical. If `TRUE`, adds a `response_scored` column with reverse-scored values applied for instruments that define `reverse_items` (currently STAI-S and STAI-T). For all other items and instruments, `response_scored` equals `response`. Non-numeric responses (e.g. clock-time lists, yes/no) are left as-is. Defaults to `FALSE` to preserve existing behaviour.
`instruments`	An optional named list of additional registry entries from `load_instrument()` or `load_instrument_dir()`, used when `scored_items = TRUE` to resolve reverse-scoring metadata for custom instruments.

Value

A `data.frame` with columns: participant metadata (optional), `questionnaire_id`, `completed_at`, `item_id`, `response`, and optionally `response_scored`.

`load_instrument` *Load a custom instrument from a ScoreMe JSON spec*

Description

Reads a `.json` file produced for (or exported from) the ScoreMe app and compiles it into a tallieR registry entry. The result can be passed to `score_questionnaire()` via its `instruments` argument, combined with other custom entries using `c()`, or stored in a project-level list for reuse.

Usage

```
load_instrument(path)
```

Arguments

`path`	Path to a `.json` file containing a ScoreMe instrument spec. See `vignette("custom-instruments` or `system.file("extdata", "example_instrument.json", package = "tallieR")` for the expected schema.

Details

Supported scoring types: `"sum"`, `"weighted_sum"`, `"mean"`. For instruments with `"composite"` scoring (e.g. PSQI-style multi-component algorithms) the returned entry will score as `NA` and a warning is emitted each time scoring is attempted. You can override the compiled `score` function by assigning a custom one after loading (see examples).

Value

A named list (a single tallieR registry), where the name is the instrument `id` from the spec. Invisibly also carries the original spec in `$<id>$spec`.

See Also

`load_instrument_dir()`, `score_questionnaire()`, `available_instruments()`

Examples

```
## Not run:
my_instr <- load_instrument("path/to/fss.json")

# Score a single result
score_questionnaire("fss", answers, instruments = my_instr)

# Combine with another custom instrument
```

```
registry <- c(load_instrument("fss.json"), load_instrument("vas_pain.json"))

# Override scoring for a composite instrument
my_psqi <- load_instrument("psqi_custom.json")
my_psqi$psqi_custom$score <- function(answers) { ... }

## End(Not run)
```

`load_instrument_dir` *Load all custom instruments from a directory*

Description

Reads every `.json` file in `dir`, compiles each as a tallieR registry entry, and returns a combined named list suitable for passing to `score_questionnaire()`.

Usage

```
load_instrument_dir(dir, pattern = "\\.json$")
```

Arguments

`dir`	Path to a directory containing ScoreMe instrument JSON files.
`pattern`	Regular expression used to filter filenames. Defaults to `"\\.json$"`.

Value

A named list of registry entries (one per successfully loaded file).

See Also

`load_instrument()`, `score_questionnaire()`

Examples

```
## Not run:
custom <- load_instrument_dir("instruments/")
score_questionnaire("fss", answers, instruments = custom)

## End(Not run)
```

`omega_reliability` *McDonald's omega for one or more questionnaires*

Description

Computes McDonald's omega (ω_t, total omega) as a measure of internal consistency for each questionnaire present in a `tallier_export` or `tallier_study` object. Omega is generally preferred over Cronbach's alpha for non-tau-equivalent items (i.e. items with unequal factor loadings), which is the norm in most psychological questionnaires.

Usage

```
omega_reliability(obj, questionnaires = NULL, min_items = 2L)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object, or a data frame as returned by `items_long()` (must contain columns `participant_id`, `questionnaire_id`, `item_id`, `completed_at`, `response`).
`questionnaires`	Character vector of questionnaire IDs to include. Defaults to all questionnaires present in `obj`.
`min_items`	Integer. Minimum number of numeric items required to attempt estimation (default `2`).

Details

Estimation uses a single-factor EFA via `stats::factanal()`. The formula applied is:

$$\omega_t = \frac{(\sum \lambda_i)^2}{(\sum \lambda_i)^2 + \sum(1 - \lambda_i^2)}$$

where λ_i are the standardised factor loadings.

Non-numeric items (e.g. MCTQ clock times, STOP-BANG yes/no) are silently dropped before estimation. Questionnaires with fewer items than observations, fewer than 2 numeric items, or non-convergent factor solutions return `NA` with an explanatory note.

Value

A `data.frame` with one row per questionnaire and columns:

questionnaire_id Questionnaire identifier.

omega McDonald's omega_t. Range: 0 to 1.

n_items Number of numeric items used.

n_obs Number of complete observations (participants).

note `NA` on success, or a short message explaining why estimation failed.

References

McDonald, R. P. (1999). *Test theory: A unified treatment.* Lawrence Erlbaum Associates.

Revelle, W., & Zinbarg, R. E. (2009). Coefficients alpha, beta, omega, and the glb: Comments on Sijtsma. *Psychometrika*, 74(1), 145–154. doi:10.1007/s113360089102z

See Also

`cronbach_alpha()`

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")

# All questionnaires
omega_reliability(study)

# Compare alpha and omega side by side
alpha <- cronbach_alpha(study, questionnaires = c("ess", "isi"))
omega <- omega_reliability(study, questionnaires = c("ess", "isi"))
merge(alpha[, c("questionnaire_id", "alpha", "n_obs")],
      omega[, c("questionnaire_id", "omega")],
      by = "questionnaire_id")

## End(Not run)
```

`read_scoreme` *Read a ScoreMe JSON export*

Description

Reads a single JSON file exported from the ScoreMe app and returns a `tallier_export` object containing participant metadata and all questionnaire results.

Usage

```
read_scoreme(path, rescore = TRUE, instruments = NULL)
```

Arguments

`path`	Path to a `.json` file exported from ScoreMe.
`rescore`	Logical. If `TRUE` (default), scores are recomputed from item-level answers using the built-in tallieR scoring functions. If `FALSE`, the scores stored in the export file are used as-is.
`instruments`	An optional named list of custom instrument registry entries created by `load_instrument()` or `load_instrument_dir()`. When `rescore = TRUE`, these are merged with the built-in registry so that custom questionnaires in the export can be scored. Entries in `instruments` take precedence over built-ins with the same id.

Value

A `tallier_export` object: a list with elements:

`exported_at`	Timestamp of the export (character).
`export_version`	Schema version string.
`participants`	A list of parsed participant records.
`n_participants`	Number of participants.

See Also

`load_instrument()`, `load_instrument_dir()`

Examples

```
## Not run:
exp <- read_scoreme("my_study_export.json")
print(exp)
wide <- scores_wide(exp)

# With custom instruments
custom <- load_instrument("fss.json")
exp    <- read_scoreme("export.json", instruments = custom)
wide   <- scores_wide(exp)

## End(Not run)
```

`read_scoreme_dir` *Read a directory of ScoreMe JSON exports*

Description

Reads all `.json` files in a directory and combines them into a single `tallier_study` object. Useful when each participant's data was exported as a separate file, or when multiple batch exports need to be merged.

Usage

```
read_scoreme_dir(
  dir,
  rescore = TRUE,
  pattern = "\\.json$",
  instruments = NULL
)
```

Arguments

`dir`	Path to a directory containing `.json` export files.
`rescore`	Logical. Passed to `read_scoreme()`.
`pattern`	Regular expression used to filter filenames. Defaults to `"\\.json$"` (all JSON files).
`instruments`	An optional named list of custom instrument registry entries, passed through to `read_scoreme()`. See `load_instrument()`.

Value

A `tallier_study` object: a list with elements:

`files` Character vector of files read.

`participants` Combined list of all participant records.

`n_participants` Total number of participants.

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")
wide  <- scores_wide(study)

## End(Not run)
```

`score_all` *Score all questionnaires in an export*

Description

Convenience wrapper that rescores every result entry in a `tallier_export` or `tallier_study` object. This is called automatically when `rescore = TRUE` in `read_scoreme()`.

Usage

```
score_all(obj, instruments = NULL)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`instruments`	An optional named list of additional registry entries from `load_instrument()` or `load_instrument_dir()`, merged with the built-in registry before scoring. Entries in `instruments` take precedence over built-ins with the same id.

Value

The same object with scores updated in-place.

`score_questionnaire` *Score a questionnaire from item-level answers*

Description

Computes the score for a single questionnaire using the official scoring algorithm embedded in tallieR, or a custom scoring function compiled from a ScoreMe JSON spec via `load_instrument()`.

Usage

```
score_questionnaire(id, answers, instruments = NULL)
```

Arguments

`id`	Character. Questionnaire identifier (e.g. `"ess"`, `"psqi"`). See `available_instruments()` for valid IDs. For custom instruments, this must match the `id` field in the spec passed to `instruments`.
`answers`	A named list of item responses, as exported by ScoreMe. Keys are item IDs (e.g. `"ess1"`, `"psqi2"`); values are the raw responses (numeric, character `"yes"`/`"no"`, or clock-time list).
`instruments`	An optional named list of additional instrument registry entries, as returned by `load_instrument()` or `load_instrument_dir()`. These are searched first, before the built-in registry, so a custom entry can override a built-in if needed.

Details

Instruments marked `beta = TRUE` in `available_instruments()` are newer additions whose scoring has been ported from ScoreMe but has not yet been independently validated in tallieR. A warning is emitted for these instruments; suppress with `suppressWarnings()` if needed.

Value

For most instruments: a single numeric score. For composite instruments (PSQI, MCTQ, DASS-21, PANSS, WHOQOL-BREF): a named list of subscale and total scores.

See Also

`load_instrument()`, `load_instrument_dir()`

Examples

```
score_questionnaire("ess", list(ess1 = 2, ess2 = 1, ess3 = 0,
                                ess4 = 3, ess5 = 1, ess6 = 0,
                                ess7 = 2, ess8 = 1))
```

```
## Not run:
# Score using a custom instrument loaded from a JSON spec
my_instr <- load_instrument("path/to/fss.json")
score_questionnaire("fss", answers, instruments = my_instr)

## End(Not run)
```

`scores_long` *Long score table*

Description

Returns a data frame with one row per participant × questionnaire × administration (i.e. all history is retained).

Usage

```
scores_long(obj, include_meta = TRUE)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`include_meta`	Logical. If `TRUE` (default), participant metadata columns are included.

Value

A `data.frame` with columns: participant metadata (optional), `questionnaire_id`, `completed_at`, `score`.

`scores_wide` *Wide score table*

Description

Returns a data frame with one row per participant and one column per questionnaire. When a participant completed a questionnaire more than once, only the most recent administration is included (use `scores_long()` to retain all administrations).

Usage

```
scores_wide(obj, include_meta = TRUE)
```

Arguments

`obj`	A `tallier_export` or `tallier_study` object.
`include_meta`	Logical. If `TRUE` (default), participant metadata columns (code, name, age, sex, etc.) are prepended.

Value

A `data.frame` with columns: participant metadata (if requested) followed by one numeric score column per questionnaire.

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")
wide  <- scores_wide(study)
head(wide)

## End(Not run)
```

`summary.tallier_export` *Summarise a tallier_export object*

Description

Prints a structured overview of a `tallier_export`, including participant count, instruments present, per-instrument completion rates, and the date range of administrations. The summary statistics are also returned invisibly as a list for programmatic use.

Usage

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

Arguments

`object`	A `tallier_export` object.
`...`	Ignored.

Value

Invisibly, a list with elements `n_participants`, `instruments`, `completion` (a data frame), and `date_range`.

Examples

```
## Not run:
exp <- read_scoreme("export.json")
summary(exp)

# Access stats programmatically
s <- summary(exp)
s$completion
```

```
## End(Not run)
```

`summary.tallier_study` *Summarise a tallier_study object*

Description

Prints a structured overview of a `tallier_study`, including participant count, number of source files, instruments present, per-instrument completion rates, and the date range of administrations. The summary statistics are also returned invisibly as a list for programmatic use.

Usage

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

Arguments

`object`	A `tallier_study` object.
`...`	Ignored.

Value

Invisibly, a list with elements `n_participants`, `n_files`, `instruments`, `completion` (a data frame), and `date_range`.

Examples

```
## Not run:
study <- read_scoreme_dir("exports/")
summary(study)

# Access stats programmatically
s <- summary(study)
s$completion

## End(Not run)
```

Index