Package: r4subcore (via r-universe)

May 15, 2026

Title Core Contracts, Parsers, and Scoring Primitives

Version 0.1.2

Description Foundational package in the R4SUB (R for Regulatory Submission) ecosystem. Defines the core evidence table schema, parsers, indicator abstractions, and scoring primitives needed to quantify clinical submission readiness. Provides a standardized contract for ingesting heterogeneous sources (validation outputs, metadata, traceability) into a single evidence framework.

License MIT + file LICENSE

URL https://r4sub.github.io/r4subcore/, https://github.com/R4SUB/r4subcore

BugReports https://github.com/R4SUB/r4subcore/issues

Depends R (>= 4.2)

Imports cli, jsonlite, rlang, stats, utils, xml2

Suggests knitr, r4subdata, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs libxml2-dev

Repository https://r4sub.r-universe.dev

Date/Publication 2026-03-16 11:21:19 UTC

RemoteUrl https://github.com/r4sub/r4subcore

RemoteRef HEAD

RemoteSha 09e76cd1d9b1dc9b8e03632e7f4678078a7c7a45

Contents

`aggregate_indicator_score` *Aggregate Indicator Scores*

Description

Computes summary scores from an evidence table, grouped by one or more columns.

Usage

```
aggregate_indicator_score(
  ev,
  by = "indicator_id",
  method = c("mean", "min", "weighted")
)
```

Arguments

`ev`	A valid evidence data.frame.
`by`	Character vector of column names to group by. Default: `c("indicator_id")`.
`method`	Aggregation method: `"mean"`, `"min"`, or `"weighted"`. The `"weighted"` method uses `severity_to_weight()` and `result_to_score()`.

Value

A data.frame with grouping columns plus `score` (0–1) and `n_evidence` (count of rows).

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
ev <- suppressMessages(as_evidence(
  data.frame(
    asset_type = rep("validation", 3), asset_id = rep("ADSL", 3),
    source_name = rep("pinnacle21", 3),
    indicator_id = c("SD0001", "SD0001", "SD0002"),
    indicator_name = c("SD0001", "SD0001", "SD0002"),
    indicator_domain = rep("quality", 3),
    severity = c("high", "medium", "low"),
    result = c("fail", "warn", "pass"),
    stringsAsFactors = FALSE
  ),
  ctx = ctx
))
aggregate_indicator_score(ev, by = "indicator_id", method = "weighted")
```

`as_evidence` *Coerce to Evidence Table*

Description

Takes a data.frame and coerces it into a valid evidence table. Fills in missing nullable columns with `NA` of the correct type and validates controlled vocabulary columns.

Usage

```
as_evidence(x, ctx = NULL, ...)
```

Arguments

`x`	A data.frame (or tibble) with at least the required evidence columns.
`ctx`	An optional r4sub_run_context. If provided, `run_id` and `study_id` are filled from the context when missing.
`...`	Additional columns to set (e.g., `asset_type = "validation"`).

Value

A data.frame conforming to the evidence schema.

Examples

```
ctx <- r4sub_run_context("STUDY1", "DEV")
df <- data.frame(
  asset_type = "validation",
  asset_id = "ADSL",
  source_name = "pinnacle21",
  indicator_id = "P21-001",
  indicator_name = "Missing variable",
  indicator_domain = "quality",
  severity = "high",
  result = "fail",
  message = "Variable AGEU missing",
  stringsAsFactors = FALSE
)
ev <- as_evidence(df, ctx = ctx)
```

`bind_evidence` *Bind Evidence Tables*

Description

Row-binds multiple evidence data.frames after validating each one.

Usage

```
bind_evidence(...)
```

Arguments

`...`	Evidence data.frames to bind.

Value

A single combined evidence data.frame.

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
make_ev <- function(ind_id) {
  suppressMessages(as_evidence(
    data.frame(
      asset_type = "validation", asset_id = "ADSL",
      source_name = "pinnacle21", indicator_id = ind_id,
      indicator_name = ind_id, indicator_domain = "quality",
      severity = "low", result = "pass",
      stringsAsFactors = FALSE
    ),
    ctx = ctx
```

```
  ))
}
ev1 <- make_ev("IND-001")
ev2 <- make_ev("IND-002")
combined <- suppressMessages(bind_evidence(ev1, ev2))
nrow(combined)
```

`canon_result` *Canonical Result Values*

Description

Maps common result/status labels to the canonical set: `pass`, `fail`, `warn`, `na`.

Usage

```
canon_result(x)
```

Arguments

`x`	Character vector of result values.

Value

Character vector with canonical result labels.

Examples

```
canon_result(c("PASS", "Failed", "Warning", "N/A"))
```

`canon_severity` *Canonical Severity Values*

Description

Maps common severity labels (case-insensitive) to the canonical set.

Usage

```
canon_severity(x)
```

Arguments

`x`	Character vector of severity values.

Value

Character vector with canonical severity labels.

Examples

```
canon_severity(c("HIGH", "Low", "warning", "Error"))
```

`define_xml_to_evidence` *Parse Define-XML to Evidence*

Description

Reads a Define-XML 2.0/2.1 file and extracts dataset, variable, and derivation completeness checks into the standard evidence table format. Three indicators are evaluated for each asset:

Q-DEFINE-001 Dataset is present and has a non-empty label.

Q-DEFINE-002 Variable is documented (has label and dataType).

Q-DEFINE-003 Derivation text is present for derived variables.

Usage

```
define_xml_to_evidence(file, ctx, source_version = "2.1")
```

Arguments

`file`	Character. Path to a Define-XML file (`.xml`).
`ctx`	An r4sub_run_context providing run and study metadata.
`source_version`	Character. Version label for the Define-XML standard. Default: `"2.1"`.

Value

A data.frame conforming to the evidence schema, one row per dataset-level check (Q-DEFINE-001), per variable check (Q-DEFINE-002), and per derivation check (Q-DEFINE-003).

Examples

```
# Build a minimal Define-XML 2.1 document
xml_txt <- '<?xml version="1.0" encoding="UTF-8"?>
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.3"
     xmlns:def="http://www.cdisc.org/ns/def/v2.1">
  <Study OID="STUDY001">
    <MetaDataVersion OID="MDV.001" Name="Define-XML 2.1">
      <def:ItemGroupDef OID="IG.ADSL" Name="ADSL" SASDatasetName="ADSL"
                        Repeating="No" Purpose="Analysis"
                        def:Label="Subject-Level Analysis Dataset">
```

```
          <ItemRef ItemOID="IT.ADSL.USUBJID" Mandatory="Yes"/>
          <ItemRef ItemOID="IT.ADSL.AGE" Mandatory="Yes"/>
        </def:ItemGroupDef>
        <ItemDef OID="IT.ADSL.USUBJID" Name="USUBJID"
                 DataType="text" Length="20"
                 SASFieldName="USUBJID">
       <Description><TranslatedText>Unique Subject Identifier</TranslatedText></Description>
        </ItemDef>
        <ItemDef OID="IT.ADSL.AGE" Name="AGE"
                 DataType="integer" Length="8"
                 SASFieldName="AGE">
          <Description><TranslatedText>Age</TranslatedText></Description>
          <def:Origin Type="Derived">
         <def:Description><TranslatedText>Derived from RFSTDTC</TranslatedText></def:Description>
          </def:Origin>
        </ItemDef>
      </MetaDataVersion>
    </Study>
</ODM>'
tmp <- tempfile(fileext = ".xml")
writeLines(xml_txt, tmp)
ctx <- r4sub_run_context("STUDY001", "DEV")
ev <- define_xml_to_evidence(tmp, ctx)
nrow(ev)
```

`evidence_schema` *Evidence Table Schema Definition*

Description

Returns the column specification for the R4SUB evidence table. Each element describes a column's expected R type and, where applicable, the set of allowed values.

Usage

```
evidence_schema()
```

Value

A named list. Each element is a list with `type` (character) and optionally `allowed` (character vector) or `nullable` (logical).

Examples

```
str(evidence_schema())
```

evidence_summary *Summarize Evidence*

Description

Returns a summary data.frame with counts grouped by domain, severity, result, and source.

Usage

```
evidence_summary(ev)
```

Arguments

ev	A valid evidence data.frame.

Value

A data.frame with columns: `indicator_domain`, `severity`, `result`, `source_name`, and `n`.

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
ev <- suppressMessages(as_evidence(
  data.frame(
    asset_type = "validation", asset_id = "ADSL",
    source_name = "pinnacle21", indicator_id = "SD0001",
    indicator_name = "SD0001", indicator_domain = "quality",
    severity = "high", result = "fail",
    stringsAsFactors = FALSE
  ),
  ctx = ctx
))
evidence_summary(ev)
```

export_evidence *Export Evidence Table to File*

Description

Validates the evidence table then writes it to disk in the requested format. Metadata attributes (`exported_at`, `r4subcore_version`, `nrow`) are attached to the returned path value for traceability.

Usage

```
export_evidence(evidence, file, format = c("csv", "rds", "json"))
```

Arguments

evidence	A valid evidence data.frame (as produced by as_evidence()).
file	Character. Destination file path (including extension).
format	Character. One of "csv", "rds", or "json". Default: "csv".

Value

Invisibly returns file with attributes:

exported_at POSIXct timestamp of export.

r4subcore_version Package version string.

nrow Number of rows written.

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
ev <- suppressMessages(as_evidence(
  data.frame(
    asset_type = "validation", asset_id = "ADSL",
    source_name = "pinnacle21", indicator_id = "SD0001",
    indicator_name = "SD0001", indicator_domain = "quality",
    severity = "high", result = "fail",
    stringsAsFactors = FALSE
  ),
  ctx = ctx
))
tmp_csv <- tempfile(fileext = ".csv")
out <- suppressMessages(export_evidence(ev, tmp_csv, format = "csv"))
file.exists(out)
```

---

hash_id *Generate a Stable Hash ID*

Description

Creates a deterministic hash from one or more character inputs. Uses MD5 via base R's digest-like approach for a lightweight, dependency-free implementation.

Usage

```
hash_id(..., prefix = NULL)
```

Arguments

...	Character values to hash together. Concatenated with "|".
prefix	Optional prefix prepended to the hash (e.g., "RUN", "IND").

Value

A character string of the form `prefix-hexhash` or just `hexhash`.

Examples

```
hash_id("ADSL", "rule_001")
hash_id("my_study", "2024-01-01", prefix = "RUN")
```

`import_evidence` *Import Evidence Table from File*

Description

Reads an evidence table that was previously saved by `export_evidence()`, then validates it against the evidence schema.

Usage

```
import_evidence(file, format = c("csv", "rds", "json"))
```

Arguments

`file`	Character. Path to the file to read.
`format`	Character. One of `"csv"`, `"rds"`, or `"json"`. Default: `"csv"`.

Value

A validated evidence data.frame conforming to the evidence schema.

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
ev <- suppressMessages(as_evidence(
  data.frame(
    asset_type = "validation", asset_id = "ADSL",
    source_name = "pinnacle21", indicator_id = "SD0001",
    indicator_name = "SD0001", indicator_domain = "quality",
    severity = "high", result = "fail",
    stringsAsFactors = FALSE
  ),
  ctx = ctx
))
tmp_rds <- tempfile(fileext = ".rds")
suppressMessages(export_evidence(ev, tmp_rds, format = "rds"))
ev2 <- suppressMessages(import_evidence(tmp_rds, format = "rds"))
nrow(ev2)
```

json_safely *Safely Serialize to JSON String*

Description

Converts an R object to a valid JSON string. Returns `"{}"` on failure or for `NULL`/empty inputs.

Usage

```
json_safely(x)
```

Arguments

`x`	An R object to serialize.

Value

A single character string containing valid JSON.

Examples

```
json_safely(list(a = 1, b = "hello"))
json_safely(NULL)
```

normalize_01 *Normalize to 0–1 Range*

Description

Applies min-max normalization to a numeric vector, optionally clamping values to [0, 1].

Usage

```
normalize_01(x, direction = c("higher_better", "lower_better"), clamp = TRUE)
```

Arguments

`x`	Numeric vector.
`direction`	Character. `"higher_better"` (default) maps max to 1; `"lower_better"` maps min to 1.
`clamp`	Logical. If `TRUE`, clamp output to [0, 1].

Value

Numeric vector normalized to 0–1.

Examples

```
normalize_01(c(10, 20, 30, 40, 50))
normalize_01(c(10, 20, 30), direction = "lower_better")
```

`p21_to_evidence` *Parse Pinnacle21 Output to Evidence*

Description

Converts a data.frame of Pinnacle21-style validation results into the standard evidence table format. Column names are detected case-insensitively.

Usage

```
p21_to_evidence(
  p21_df,
  ctx,
  asset_type = "validation",
  source_version = NULL,
  default_domain = "quality"
)
```

Arguments

`p21_df`	A data.frame containing Pinnacle21 validation output. Expected columns (case-insensitive): `Rule` (or `Rule ID`), `Message`, `Severity`, `Dataset`, `Variable`, `Result` (or `Status`).
`ctx`	A r4sub_run_context providing run and study metadata.
`asset_type`	Character. Asset type label. Default: `"validation"`.
`source_version`	Character or `NULL`. Version of the P21 tool.
`default_domain`	Character. Indicator domain. Default: `"quality"`.

Value

A data.frame conforming to the evidence schema.

Examples

```
p21_raw <- data.frame(
  Rule = c("SD0001", "SD0002"),
  Message = c("Missing variable label", "Invalid format"),
  Severity = c("Error", "Warning"),
  Dataset = c("ADSL", "ADAE"),
  Variable = c("AGE", "AESTDTC"),
  Status = c("Failed", "Warning"),
  stringsAsFactors = FALSE
```

```
)
ctx <- r4sub_run_context("STUDY1", "DEV")
ev <- p21_to_evidence(p21_raw, ctx)
```

`r4sub_run_context` *Create a Run Context*

Description

A run context captures metadata for a particular evidence collection run. It provides a unique `run_id`, study identifier, environment label, and timestamps used throughout evidence ingestion.

Usage

```
r4sub_run_context(
  study_id,
  environment = c("DEV", "UAT", "PROD"),
  user = NULL,
  run_id = NULL,
  timestamp = Sys.time()
)
```

Arguments

`study_id`	Character. Study identifier (e.g., `"ABC123"`).
`environment`	Character. One of `"DEV"`, `"UAT"`, `"PROD"`.
`user`	Character or `NULL`. Username; defaults to system user.
`run_id`	Character or `NULL`. If `NULL`, a unique ID is generated.
`timestamp`	POSIXct. Defaults to current time.

Value

A list of class `r4sub_run_context` with elements: `run_id`, `study_id`, `environment`, `user`, `created_at`.

Examples

```
ctx <- r4sub_run_context(study_id = "STUDY001", environment = "DEV")
ctx$run_id
ctx$study_id
```

`register_indicator` *Register an Indicator*

Description

Adds an indicator definition to the local in-memory registry.

Usage

```
register_indicator(
  indicator_id,
  domain,
  description,
  expected_inputs = character(0),
  default_thresholds = numeric(0),
  tags = character(0)
)
```

Arguments

`indicator_id`	Character. Stable identifier for the indicator.
`domain`	Character. One of `"quality"`, `"trace"`, `"risk"`, `"usability"`.
`description`	Character. Human-readable description.
`expected_inputs`	Character vector. Evidence source types this indicator expects.
`default_thresholds`	Named numeric vector. Optional thresholds.
`tags`	Character vector. Optional tags (e.g., `"define"`, `"adam"`).

Value

The indicator definition list, invisibly.

Examples

```
register_indicator(
  indicator_id = "P21-001",
  domain = "quality",
  description = "Required variable is missing from dataset"
)
```

result_to_score *Map Result to Numeric Score*

Description

Converts canonical result labels to numeric scores.

Usage

```
result_to_score(result)
```

Arguments

result	Character vector of canonical result values (pass, fail, warn, na).

Value

Numeric vector: pass=1, warn=0.5, fail=0, na=NA.

Examples

```
result_to_score(c("pass", "fail", "warn", "na"))
```

severity_to_weight *Map Severity to Numeric Weight*

Description

Converts canonical severity labels to numeric penalty multipliers on a 0–1 scale.

Usage

```
severity_to_weight(severity)
```

Arguments

severity	Character vector of canonical severity values (info, low, medium, high, critical).

Details

Default mapping:

- info = 0.00
- low = 0.25
- medium = 0.50
- high = 0.75
- critical = 1.00

Value

Numeric vector of weights.

Examples

```
severity_to_weight(c("low", "high", "critical"))
```

`validate_evidence` *Validate Evidence Table*

Description

Checks that a data.frame conforms to the evidence schema. Verifies column presence, types, and controlled vocabulary values.

Usage

```
validate_evidence(ev)
```

Arguments

`ev`	A data.frame to validate.

Value

`TRUE` invisibly if valid; throws an error otherwise.

Examples

```
ctx <- suppressMessages(r4sub_run_context("STUDY1", "DEV"))
ev <- suppressMessages(as_evidence(
  data.frame(
    asset_type = "validation", asset_id = "ADSL",
    source_name = "pinnacle21", indicator_id = "SD0001",
    indicator_name = "SD0001", indicator_domain = "quality",
    severity = "high", result = "fail",
    stringsAsFactors = FALSE
  ),
  ctx = ctx
))
validate_evidence(ev)
```

validate_indicator *Validate Indicator Metadata*

Description

Checks that an indicator definition list is well-formed.

Usage

```
validate_indicator(indicator)
```

Arguments

indicator	A list with required fields: indicator_id, domain, description. Optional fields: expected_inputs, default_thresholds, tags.

Value

TRUE invisibly if valid; throws an error otherwise.

Examples

```
validate_indicator(list(
  indicator_id = "P21-001",
  domain = "quality",
  description = "Missing required variable"
))
```

Index