

Package ‘twbparser’

September 28, 2026

Title Parse 'Tableau' Workbooks into Functional Data

Version 0.5.1

Description High-performance parsing of 'Tableau' workbook files into tidy data frames and dependency graphs for other visualization tools like R 'Shiny' or 'Power BI' replication, plus an interactive 'Shiny' workbook inspector for uploaded .twb and .twbx files.

License MIT + file LICENSE

URL <https://prigasg.github.io/twbparser/>,
<https://github.com/PrigasG/twbparser>

BugReports <https://github.com/PrigasG/twbparser/issues>

Depends R (>= 4.2.0)

Imports dplyr, igraph, purrr, R6, rlang, stringr, tibble, tidyr,
withr, xml2

Suggests cli, covr, ggraph, knitr, lintr, magrittr, optparse,
rmarkdown, shiny, spelling, styler, testthat (>= 3.0.0),
tidygraph, vctrs, zip

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

NeedsCompilation no

Author George Arthur [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-1975-1459>>)

Maintainer George Arthur <prigasgenthian48@gmail.com>

Repository CRAN

Date/Publication 2026-09-28 17:50:02 UTC

Contents

audit_tableau_folder	3
build_dependency_graph	4
export_migration_bundle	5
extract_calculated_fields	6
extract_columns_with_table_source	7
extract_datasource_details	8
extract_joins	9
extract_named_connections	10
extract_parameters	11
extract_raw_fields	12
extract_relations	13
extract_relationships	14
extract_twbl_from_twbl	15
infer_implicit_relationships	15
parse_twbl	16
plot_dependency_graph	17
plot_relationship_graph	18
plot_source_join_graph	18
prettify_calculated_fields	19
render_migration_brief	20
run_twblparser_app	20
scaffold_quarto_dashboard	21
scaffold_shiny_dashboard	21
tableau_formula_pretty	22
translate_tableau_calc	22
TwblParser	23
twbl_extract_files	27
twbl_list	27
twbl_calc_build_order	28
twbl_calc_complexity	29
twbl_charts	30
twbl_colors	31
twbl_compatibility	31
twbl_custom_sql	32
twbl_dashboards	33
twbl_dashboard_actions	33
twbl_dashboard_filters	34
twbl_dashboard_layout	35
twbl_dashboard_sheets	36
twbl_dashboard_size	37
twbl_dashboard_summary	38
twbl_field_usage	38
twbl_formatting	39
twbl_initial_sql	40
twbl_lineage	41
twbl_migration_assessment	42

<i>audit_tableau_folder</i>	3
twb_pages	42
twb_pages_summary	43
twb_page_composition	43
twb_parameter_usage	44
twb_published_refs	45
twb_replication_brief	46
twb_sheet_axes	47
twb_sheet_filters	48
twb_sheet_shelves	49
twb_sheet_sorts	50
twb_sheet_spec	51
twb_tooltips	53
twb_unused_fields	54
validate_relationships	55
Index	57

<i>audit_tableau_folder</i>	<i>Audit a folder of Tableau workbooks</i>
-----------------------------	--

Description

Batch parses .twb and .twbx files and returns migration-oriented inventory tables. The summary table is one row per workbook; detail tables retain workbook identity so they can be filtered or joined downstream.

Usage

```
audit_tableau_folder(  
  path,  
  recursive = TRUE,  
  pattern = NULL,  
  write_csv = FALSE,  
  output_dir = NULL  
)
```

Arguments

path	Folder containing Tableau workbooks, or a character vector of .twb / .twbx files.
recursive	Logical; search folders recursively. Default TRUE.
pattern	Optional regular expression applied to file basenames.
write_csv	Logical; write each returned table as a CSV file. Default FALSE.
output_dir	Directory for CSV output when write_csv = TRUE.

Value

A named list with class `twbparser_audit` containing:

workbooks One row per parsed workbook with counts and migration complexity flags.

datasources Datasource inventory across parsed workbooks.

calculated_fields Calculated field complexity across workbooks.

field_usage Worksheet field usage across workbooks.

lineage_edges Combined lineage edge table across workbooks.

issues Files that could not be parsed and their error messages.

Examples

```
demo <- system.file("extdata", package = "twbparser")
if (nzchar(demo)) {
  audit <- audit_tableau_folder(demo)
  audit$workbooks
}
```

`build_dependency_graph`

Build a field dependency graph from calculated fields

Description

Creates a directed graph where edges point from input fields used in a formula to the calculated output field. Tokens are extracted from bracketed references like `[Table].[Field]` or `[Field]`.

Usage

```
build_dependency_graph(fields_df)
```

Arguments

`fields_df` A data frame with at least columns `name` and `formula`.

Value

An `igraph` directed graph where vertices are field names and edges represent dependencies (input -> output).

Examples

```
fields <- tibble::tibble(
  name = c("X_plus_Y", "Z"),
  formula = c("[X] + [Y]", "[X_plus_Y] * 2")
)
g <- build_dependency_graph(fields)
```

export_migration_bundle

Export a Tableau migration bundle

Description

Writes a folder of migration artifacts such as inventory CSVs, lineage, compatibility results, formula translation candidates, and a Markdown brief.

Usage

```
export_migration_bundle(
  x,
  target = c("powerbi", "shiny", "quarto", "looker", "superset"),
  path = "twbparser-migration-bundle",
  include_scaffold = TRUE
)
```

Arguments

x	A TwbParser object or path to a .twb / .twbx file.
target	Target tool. One of "powerbi", "shiny", "quarto", "looker", or "superset".
path	Output directory.
include_scaffold	Logical; include a Shiny or Quarto scaffold when target is "shiny" or "quarto". Default TRUE.

Value

Invisibly returns a tibble of written files.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  out <- file.path(tempdir(), "twbparser-bundle")
  export_migration_bundle(parser, target = "shiny", path = out)
}
```

 extract_calculated_fields

Extract calculated fields from a TWB

Description

Finds columns that contain <calculation>nodes and returns metadata and formulas, with a heuristic flag for table calculations.

Usage

```
extract_calculated_fields(xml_doc, include_parameters = FALSE)
```

Arguments

xml_doc An xml2 document for a Tableau .twb.
include_parameters Logical. If TRUE, include items from the "Parameters" datasource or columns with @param-domain-type. Default FALSE.

Value

A tibble with columns:

datasource Datasource name.
name User-visible caption or cleaned internal name.
tableau_internal_name Internal Tableau name (often bracketed).
datatype Tableau datatype.
role Tableau role.
formula Calculation formula string.
calc_class Tableau calc class (often "tableau").
is_table_calc Heuristic flag for table calcs (e.g., WINDOW_, LOOKUP).
table Raw table reference.
table_clean Cleaned table name.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
calcs <- extract_calculated_fields(xml)
head(calcs)
```

`extract_columns_with_table_source`*Extract columns with their source tables from a TWB*

Description

Scans top-level <datasource> nodes (excluding view-specific references) and returns fields with raw names/captions, cleaned table/field names, and basic metadata.

Usage

```
extract_columns_with_table_source(xml_doc)
```

Arguments

`xml_doc` An xml2 document for a Tableau .twb.

Value

A tibble with columns:

datasource Datasource name.

name Raw column name (may include brackets/qualifiers).

caption Column caption if present.

datatype Tableau datatype.

role Tableau role.

semantic_role Semantic role if present.

table Raw table reference.

table_clean Cleaned table name (no brackets/suffix).

field_clean Cleaned field name.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
fields <- extract_columns_with_table_source(xml)
```

extract_datasource_details

Extract datasource details from a Tableau TWB

Description

Gathers runtime tables (from the object graph), merges in named-connection metadata (class, caption, targets), and augments with top-level datasource definitions (field counts, connection type, location). Also returns the workbook's parameter fields via [extract_parameters\(\)](#).

Usage

```
extract_datasource_details(xml_doc)
```

Arguments

`xml_doc` An xml2 document for a Tableau .twb.

Value

A named list with:

data_sources Tibble of datasources joined with connection metadata.

parameters Tibble of parameter fields from [extract_parameters\(\)](#).

all_sources Same as data_sources (placeholder for future variants).

Examples

```
# Preferred: from a tiny .twb
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  xml <- xml2::read_xml(twb)
  res <- extract_datasource_details(xml)
  head(res$data_sources)
}
```

```
# Alternative: from a tiny .twbx (guarded)
twbx <- system.file("extdata", "test_for_zip.twbx", package = "twbparser")
if (nzchar(twbx) && file.exists(twbx)) {
  members <- twbx_list(twbx)
  twb_rows <- members$name[grepl("\\.twb$", members$name)]
  if (length(twb_rows) > 0L && !is.na(twb_rows[1])) {
    twb_member <- twb_rows[1]
    xml <- xml2::read_xml(utils::unzip(twbx, twb_member, exdir = tempdir()))
    res <- extract_datasource_details(xml)
    head(res$data_sources)
  }
}
```

extract_joins	<i>Extract Tableau join clauses from <relation type="join"> nodes</i>
---------------	---

Description

Handles both column-based clauses (<clause><column/></clause>) and expression-based predicates (<expression op=...>) found in TWB XML.

Usage

```
extract_joins(xml_doc)
```

Arguments

xml_doc An xml2 document for a Tableau .twb.

Value

A tibble with columns:

join_type Join kind (e.g., inner, left), if available.

left_table Left table name (cleaned).

left_field Left field name.

operator Predicate operator (defaults to "=" when missing).

right_table Right table name (cleaned).

right_field Right field name.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
extract_joins(xml)
```

`extract_named_connections`*Extract <named-connection> entries from a TWB*

Description

Rich, safe extraction of <named-connection> nodes and their <connection> attributes into a tidy tibble.

Usage

```
extract_named_connections(xml_doc)
```

Arguments

`xml_doc` An xml2 document for a Tableau .twb.

Value

Tibble with columns like `connection_id`, `connection_caption`, `connection_class`, `connection_target`, `dbname`, `schema`, `warehouse`, `region`, `filename`, and `location_named`.

Examples

```
# Preferred: read from a tiny '.twb'
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  xml <- xml2::read_xml(twb)
  extract_named_connections(xml)
}

twbx <- system.file("extdata", "test_for_zip.twbx", package = "twbparser")
if (nzchar(twbx) && file.exists(twbx)) {
  members <- twbx_list(twbx)
  twb_rows <- members$name[grepl("\\.twb$", members$name)]
  if (length(twb_rows) > 0L && !is.na(twb_rows[1])) {
    twb_member <- twb_rows[1]
    xml <- xml2::read_xml(utils::unzip(twbx, twb_member, exdir = tempdir()))
    extract_named_connections(xml)
  }
}
```

extract_parameters	<i>Extract parameter fields from a TWB</i>
--------------------	--

Description

Returns parameter columns (those with param-domain-type) and basic metadata, including a best-effort current value if present.

Usage

```
extract_parameters(xml_doc)
```

Arguments

xml_doc An xml2 document for a Tableau .twb.

Value

A tibble with columns:

datasource Datasource name.
name User-visible caption or cleaned internal name.
tableau_internal_name Internal Tableau name.
datatype Tableau datatype.
role Tableau role.
parameter_type Tableau parameter domain type.
allowable_type Underlying data-type (if present).
current_value Current value if specified.
is_parameter Always TRUE.
table Raw table reference.
table_clean Cleaned table name.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
params <- extract_parameters(xml)
head(params)
```

extract_raw_fields	<i>Extract non-calculated, non-parameter fields from a TWB</i>
--------------------	--

Description

Returns raw columns excluding calculated fields and parameters.

Usage

```
extract_raw_fields(xml_doc)
```

Arguments

`xml_doc` An xml2 document for a Tableau .twb.

Value

A tibble with columns:

datasource Datasource name.

name User-visible caption or cleaned internal name.

tableau_internal_name Internal Tableau name.

datatype Tableau datatype.

role Tableau role.

is_hidden Whether the field is hidden.

is_parameter Always FALSE.

table Raw table reference.

table_clean Cleaned table name.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
raw_fields <- extract_raw_fields(xml)
head(raw_fields)
```

extract_relations	<i>Extract all <relation> tags from a TWB</i>
-------------------	---

Description

Returns a tibble of <relation> elements found in a Tableau TWB XML, with key attributes and any custom SQL text.

Usage

```
extract_relations(xml_doc)
```

Arguments

xml_doc	An xml2 document for a Tableau .twb.
---------	--------------------------------------

Value

A tibble with columns:

name	Relation name
table	Table reference
connection	Connection ID
type	Relation type (table, join, etc.)
join	Join type if applicable
custom_sql	Inline SQL text if present

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
fields <- extract_columns_with_table_source(xml)
inferred <- infer_implicit_relationships(fields)
head(inferred)
```

`extract_relationships` *Extract modern relationships from a Tableau TWB*

Description

Parses Tableau "relationships" (introduced in 2020.2) between logical tables, including the join predicate fields and operator.

Usage

```
extract_relationships(xml_doc)
```

Arguments

`xml_doc` An xml2 document for a Tableau .twb.

Value

A tibble with columns:

<code>relationship_type</code>	Always "Relationship"
<code>left_table</code>	Left table name
<code>right_table</code>	Right table name
<code>left_field</code>	Field name on left side
<code>operator</code>	Join operator (e.g., "=")
<code>right_field</code>	Field name on right side
<code>left_is_calc</code>	Logical, whether left field is a calculation
<code>right_is_calc</code>	Logical, whether right field is a calculation

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
extract_relationships(xml)
```

extract_twb_from_twbx *Extract the .twb (and optionally all files) from a .twbx*

Description

Extract the .twb (and optionally all files) from a .twbx

Usage

```
extract_twb_from_twbx(
  twbx_path,
  extract_dir = file.path(tempdir(), paste0("twbx_",
    tools::file_path_sans_ext(basename(twbx_path)), "_", format(Sys.time(),
      "%Y%m%d%H%M%S"))),
  extract_all = FALSE
)
```

Arguments

twbx_path	Path to a .twbx file.
extract_dir	Directory to extract into (defaults to a timestamped temp dir).
extract_all	If TRUE, extract entire archive; otherwise only the largest .twb.

Value

List with twb_path, exdir, twbx_path, and manifest (tibble).

Examples

```
twbx <- system.file("extdata", "test_for_zip.twbx", package = "twbparser")
res <- extract_twb_from_twbx(twbx, extract_all = FALSE)
basename(res$twb_path)
```

infer_implicit_relationships

Infer implicit relationships between tables from field metadata

Description

Generates candidate join pairs by:

- Matching semantic_role across different tables.
- Matching field names (case-insensitive) across different tables.

Usage

```
infer_implicit_relationships(fields_df, max_pairs = 50000L)
```

Arguments

fields_df A data frame like the output of `extract_columns_with_table_source()`.
max_pairs Maximum number of candidate pairs to return (default 50,000).

Value

A tibble with columns:

left_table Left table name.

left_field Left field name.

right_table Right table name.

right_field Right field name.

reason Why the pair was suggested.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
fields <- extract_columns_with_table_source(xml)
inferred <- infer_implicit_relationships(fields)
head(inferred)
```

parse_twb

Parse a Tableau workbook and write a batch export to disk

Description

`parse_twb()` is a convenience wrapper for non-interactive use: it parses a .twb/.twbx workbook with [TwbParser](#) and writes a structured set of outputs into `output_dir` — a human-readable report, one CSV per key table (including the rebuild kit: unused fields, calculation build order, and parameter usage), per-worksheet visualization specs, a plain-text replication brief, and the field dependency graph as GraphML (readable with `igraph/ggraph` or any GraphML tool).

Usage

```
parse_twb(path, output_dir = "results", overwrite = FALSE, quiet = FALSE)
```

Arguments

path	Path to a .twb or .twbx file.
output_dir	Directory to write outputs into. Created if needed (including parents).
overwrite	If FALSE (default), refuse to write into an existing non-empty directory instead of mixing outputs. If TRUE, the previous outputs written by parse_twb in the directory are removed first so the export reflects the current workbook; unrelated files are left alone.
quiet	If TRUE, suppress progress messages.

Value

The normalized output_dir, invisibly.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  out <- parse_twb(twb, output_dir = file.path(tempdir(), "twbparser-demo"),
                  quiet = TRUE)
  list.files(out)
}
```

plot_dependency_graph *Plot a field dependency graph*

Description

Draws a quick base-graphics plot of a dependency graph. Vertices that are calculated fields (present in fields_df\$name) are drawn differently.

Usage

```
plot_dependency_graph(g, fields_df = NULL, seed = NULL)
```

Arguments

g	An igraph directed graph from build_dependency_graph() .
fields_df	Optional data frame with a name column to mark calculated outputs.
seed	Optional integer seed to make the layout reproducible. If NULL (default), the function will not alter the caller's RNG state.

Value

Invisibly returns g.

Examples

```
fields <- tibble::tibble(
  name = c("X_plus_Y", "Z"),
  formula = c("[X] + [Y]", "[X_plus_Y] * 2")
)
g <- build_dependency_graph(fields)
plot_dependency_graph(g, fields)           # nondeterministic layout
plot_dependency_graph(g, fields, seed = 1) # deterministic layout
```

```
plot_relationship_graph
```

Plot a field-level relationship DAG (legacy)

Description

Uses relationships_df with columns left_table, right_table, left_field, right_field, and optional operator.

Usage

```
plot_relationship_graph(relationships_df, seed = NULL)
```

Arguments

relationships_df	Data frame of field-level relationships.
seed	Optional integer seed to make the layout reproducible. If NULL (default), the function preserves the caller's RNG state.

Value

Invisibly returns the plotted graph.

```
plot_source_join_graph
```

Plot a source join graph

Description

Visualizes joins between sources. Expects joins_df with columns left_table, right_table, left_field, right_field. If relationships_df is provided (modern relationships), it will render a second graph highlighting those relationships.

Usage

```
plot_source_join_graph(joins_df, relationships_df = NULL, seed = NULL)
```

Arguments

`joins_df` Data frame with join edges.

`relationships_df` Optional data frame with modern relationships.

`seed` Optional integer seed to make layouts reproducible. If NULL (default), the function preserves the caller's RNG state.

Value

Invisibly returns the join graph, or a list `list(joins = g, relationships = gr)` if `relationships_df` is provided.

```
prettify_calculated_fields
```

Add a prettified formula column to calculated fields table

Description

Add a prettified formula column to calculated fields table

Usage

```
prettify_calculated_fields(calcs, strip_brackets = FALSE, wrap = 100L)
```

Arguments

`calcs` tibble from `extract_calculated_fields()`

`strip_brackets` logical

`wrap` integer wrap width; default 100

Value

tibble with extra column `formula_pretty`

`render_migration_brief`*Render a migration brief*

Description

Creates a Markdown migration brief from parsed workbook metadata and optional target-specific compatibility notes.

Usage

```
render_migration_brief(  
  x,  
  target = c("powerbi", "shiny", "quarto", "looker", "superset"),  
  output = NULL  
)
```

Arguments

<code>x</code>	A TwbParser object or an xml2 document.
<code>target</code>	Target tool.
<code>output</code>	Optional file path. When supplied, the brief is written to disk.

Value

A character scalar containing Markdown.

`run_twbparser_app`*Launch the twbparser Shiny app*

Description

Opens an interactive workbook inspector for .twb and .twbx files.

Usage

```
run_twbparser_app(...)
```

Arguments

<code>...</code>	Additional arguments passed to shiny::runApp() .
------------------	--

Value

The value returned by [shiny::runApp\(\)](#).

`scaffold_quarto_dashboard`*Scaffold a Quarto dashboard rebuild*

Description

Writes a minimal Quarto dashboard with one section per Tableau dashboard and worksheet placeholders.

Usage

```
scaffold_quarto_dashboard(x, path = "quarto-scaffold")
```

Arguments

<code>x</code>	A TwbParser object or an xml2 document.
<code>path</code>	Output directory.

Value

Path to the written .qmd file.

`scaffold_shiny_dashboard`*Scaffold a Shiny dashboard rebuild*

Description

Writes a minimal app.R with one tab per Tableau dashboard and placeholders for worksheets detected in the workbook.

Usage

```
scaffold_shiny_dashboard(x, path = "shiny-scaffold")
```

Arguments

<code>x</code>	A TwbParser object or an xml2 document.
<code>path</code>	Output directory.

Value

Path to the written app.R.

tableau_formula_pretty

Prettify a Tableau calculation formula for display

Description

Prettify a Tableau calculation formula for display

Usage

```
tableau_formula_pretty(formula, strip_brackets = FALSE, wrap = NA_integer_)
```

Arguments

formula	character scalar
strip_brackets	logical; remove [] around field names (default FALSE) []: R:%20
wrap	optional integer to hard-wrap lines (use NA to disable)

Value

character scalar (multi-line, indented)

translate_tableau_calc

Translate simple Tableau calculated fields

Description

Performs deterministic, best-effort formula rewrites for common Tableau functions. Complex Tableau features such as LOD expressions and table calculations are flagged for manual review.

Usage

```
translate_tableau_calc(formula, target = c("dax", "sql", "r"))
```

Arguments

formula	Character vector of Tableau formulas.
target	Target language: "dax", "sql", or "r".

Value

A tibble with source formula, translated formula, confidence, and review notes.

Examples

```
translate_tableau_calc("IF [Sales] > 0 THEN [Profit] ELSE 0 END", target = "sql")
```

TwbParser

Tableau Workbook Parser (R6)

Description

Create a parser for Tableau .twb / .twbx files. On initialization, the parser reads the XML and precomputes relationships, joins, fields, calculated fields, inferred relationships, and datasource details. For .twbx, it also extracts the largest .twb and records a manifest.

Format

An R6 class generator.

Details

Results are read through **active-binding properties** (no parentheses), e.g. `parser$summary`, `parser$overview`, `parser$datasources`. The `get_*`() methods return the same data and are useful when you need to pass arguments (e.g. `parser$get_sheet_shelves("My Sheet")`) or want an explicit call.

Fields

path Path to the .twb or .twbx file on disk.

xml_doc Parsed xml2 document of the workbook.

twbx_path Original .twbx path if the workbook was packaged.

twbx_dir Directory where the .twbx was extracted.

twbx_manifest Tibble of .twbx contents from `twbx_list()`.

relations Tibble of <relation> nodes from `extract_relations()`.

joins Tibble of join clauses from `extract_joins()`.

relationships Tibble of modern relationships from `extract_relationships()`.

inferred_relationships Tibble of inferred relationship pairs by name and role.

datasource_details List containing data_sources, parameters, and all_sources.

fields Tibble of raw fields with table information.

calculated_fields Tibble of calculated fields.

custom_sql Tibble of custom SQL relations from `twb_custom_sql()`.

initial_sql Tibble of initial SQL statements from `twb_initial_sql()`.

published_refs Tibble of published-datasource references from `twb_published_refs()`.

last_validation Result from `validate()` as list with ok and issues elements.

Active bindings (read-only properties)

summary The workbook report (a `twbparser_report`); accessing it prints the summary. Note: this is a property, not a method — use `parser$summary`, not `parser.summary()`.

report Same as `summary`: the full structured workbook report.

overview One-row tibble with counts of datasources, parameters, relationships, fields, dashboards, and filters.

pages Tibble of workbook pages (worksheets, dashboards, stories).

pages_summary Tibble summarizing page counts by type.

charts Tibble of chart/mark information per worksheet.

colors Tibble of color encodings used across worksheets.

dashboards Tibble of dashboards in the workbook.

dashboard_summary Tibble summarizing dashboards and their filters.

dashboard_filters Tibble of dashboard filter configurations.

datasources Tibble of datasource details (see `get_datasources()`).

parameters_tbl Tibble of parameter fields (see `get_parameters()`).

datasources_all Tibble of all sources (see `get_datasources_all()`).

fields_tbl Tibble of raw fields (see `get_fields()`).

custom_sql_tbl Tibble of custom SQL (see `get_custom_sql()`).

initial_sql_tbl Tibble of initial SQL (see `get_initial_sql()`).

published_refs_tbl Tibble of published references (see `get_published_refs()`).

twbx_manifest_tbl Tibble of .twbx contents (see `get_twbx_manifest()`).

twbx_extracts_tbl Tibble of .twbx extract entries (see `get_twbx_extracts()`).

twbx_images_tbl Tibble of .twbx image entries (see `get_twbx_images()`).

sheet_shelves Tibble of shelf placement (see `get_sheet_shelves()`).

sheet_filters Tibble of worksheet filters (see `get_sheet_filters()`).

sheet_axes Tibble of axis configuration (see `get_sheet_axes()`).

sheet_sorts Tibble of sort directives (see `get_sheet_sorts()`).

sheet_spec Named list of per-worksheet visualization specs (see `get_sheet_spec()`).

dashboard_sheets Tibble of worksheets per dashboard (see `get_dashboard_sheets()`).

dashboard_layout Tibble of the zone layout tree (see `get_dashboard_layout()`).

dashboard_actions Tibble of dashboard actions (see `get_dashboard_actions()`).

dashboard_charts Tibble of charts placed on dashboards (see `get_dashboard_charts()`).

calc_complexity Tibble of calculated-field complexity (see `get_calc_complexity()`).

field_usage Tibble of field usage across worksheets (see `get_field_usage()`).

unused_fields Tibble of defined-but-never-used fields (see `get_unused_fields()`).

calc_build_order Tibble of calculated fields in rebuild order (see `get_calc_build_order()`).

parameter_usage Tibble of parameter consumption points (see `get_parameter_usage()`).

lineage Migration-oriented datasource-to-dashboard lineage (see `get_lineage()`).

compatibility Target-tool compatibility matrix (see `get_compatibility()`).

migration_assessment Target-specific migration readiness assessment (see `get_migration_assessment()`).

validation Last validation result; runs `validate()` first if it has never been run.

Properties are cached after first access; most return the same tibbles as the corresponding `get_*`() methods.

Methods

new(path) Create a parser from a `.twb` or `.twbx` file.

get_twbx_manifest() Return `.twbx` manifest tibble.

get_twbx_extracts() Return `.twbx` extract entries.

get_twbx_images() Return `.twbx` image entries.

extract_twbx_assets(types = NULL, pattern = NULL, files = NULL, exdir = NULL) Extract files from the `.twbx` archive to disk.

get_relations() Return relations tibble.

get_joins() Return joins tibble.

get_relationships() Return modern relationships tibble.

get_inferred_relationships() Return inferred relationship pairs.

get_datasources() Return datasource details tibble.

get_parameters() Return parameters tibble.

get_datasources_all() Return all sources tibble.

get_fields() Return raw fields tibble.

get_calculated_fields(pretty = FALSE, strip_brackets = FALSE, wrap = 100L, include_parameters = FALSE)
Return calculated fields tibble. When `pretty = TRUE`, includes a `formula_pretty` column with line breaks and indentation.

get_custom_sql() Return custom SQL tibble.

get_initial_sql() Return initial SQL tibble.

get_published_refs() Return published-datasource references tibble.

get_pages() Return workbook pages tibble.

get_pages_summary() Return page counts by type.

get_page_composition(name) Return zone/mark composition of one page.

get_charts() Return chart/mark information per worksheet.

get_colors() Return color encodings used across worksheets.

get_dashboards() Return dashboards tibble.

get_dashboard_filters(dashboard = NULL) Return dashboard filter configurations.

get_dashboard_summary() Return dashboard summary tibble.

get_sheet_shelves(sheet = NULL) Fields placed on visual shelves for one or all worksheets.

get_sheet_filters(sheet = NULL) Detailed filter configuration for one or all worksheets.

get_sheet_axes(sheet = NULL) Axis configuration for one or all worksheets.

get_sheet_sorts(sheet = NULL) Sort directives for one or all worksheets.

get_sheet_spec(sheet = NULL) Full visualization spec for one or all worksheets (mark type, shelves, dimensions/measures, encodings, tooltips, filters, sorts, axes).

get_dashboard_sheets(dashboard = NULL) Worksheets embedded in one or all dashboards.

get_dashboard_layout(dashboard = NULL) Full zone layout with container hierarchy.

get_dashboard_actions(dashboard = NULL) Dashboard and workbook actions.

get_dashboard_charts(dashboard = NULL) One row per worksheet placed on a dashboard: mark type, fields, tooltip summary, and layout position.

get_dashboard_size(dashboard = NULL) Dashboard page size and sizing mode.

get_formatting(scope = NULL) Formatting rules (fonts, colours, number formats, ...); scope is one of "worksheet", "dashboard", or "workbook".

get_tooltips(sheet = NULL) Plain-text worksheet tooltips.

get_calc_complexity(include_parameters = FALSE) Calculated field complexity classifications.

get_field_usage(include_filters = TRUE, include_shelves = TRUE, wide = FALSE) Field usage matrix across worksheets.

get_unused_fields() Fields defined but never used anywhere in the workbook.

get_calc_build_order() Calculated fields topologically sorted for rebuilding.

get_parameter_usage() Where each parameter is consumed (formulas, shelves, filters).

get_lineage(format = c("tables", "igraph", "mermaid"), include_calc_dependencies = TRUE) Return migration-oriented datasource-to-dashboard lineage.

get_migration_assessment(target = c("powerbi", "shiny", "quarto", "looker", "superset")) Return a target-specific migration readiness assessment.

get_compatibility(targets = c("powerbi", "shiny", "quarto", "looker", "superset")) Return a target-tool compatibility matrix.

get_replication_brief(dashboard = NULL, include_sql = TRUE, include_formulas = TRUE, format = c("list", "text")) Full replication brief for the workbook or a single dashboard.

get_workbook_report() Return the full structured workbook report.

get_overview() Return the one-row overview tibble.

validate(error = FALSE) Validate relationships. Stops execution if error = TRUE.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb)) {
  parser <- TwbParser$new(twb)
  parser$overview
  parser$get_calculated_fields()
}
```

twbx_extract_files	<i>Extract specific files from a .twbx</i>
--------------------	--

Description

Extract specific files from a .twbx

Usage

```
twbx_extract_files(  
  twbx_path,  
  files = NULL,  
  pattern = NULL,  
  types = NULL,  
  exdir = NULL  
)
```

Arguments

twbx_path	Path to a .twbx.
files	Vector of archive paths to extract (optional).
pattern	Perl regex to match archive paths (optional).
types	Subset by .twbx entry type (see twbx_list()) (optional).
exdir	Output directory (defaults to temp).

Value

Tibble with name, type, and out_path of extracted files.

Examples

```
twbx <- system.file("extdata", "test_for_zip.twbx", package = "twbparser")  
files <- twbx_extract_files(twbx, types = c("workbook"))  
head(files)
```

twbx_list	<i>List contents of a Tableau .twbx</i>
-----------	---

Description

List contents of a Tableau .twbx

Usage

```
twbx_list(twbx_path)
```

Arguments

`twbx_path` Path to a .twbx file.

Value

Tibble with columns: name, size_bytes, modified, type.

Examples

```
twbx <- system.file("extdata", "test_for_zip.twbx", package = "twbparser")
twbx_list(twbx)
```

twb_calc_build_order	<i>Calculated fields in rebuild dependency order</i>
----------------------	--

Description

Returns every calculated field topologically sorted so that each field appears *after* the calculated fields its formula depends on. Recreate the fields in `build_order` sequence and every reference will already exist. Fields caught in a dependency cycle cannot be ordered: they get `build_order = NA` and `is_cyclic = TRUE` (plus a warning naming them).

Usage

```
twb_calc_build_order(x)
```

Arguments

`x` A TwbParser object or an xml2 document.

Details

A formula token is treated as a calc-on-calc dependency only when it matches a calculated field *and not* a raw field or parameter, so `SUM([Sales])` inside a calc captioned "Sales" does not create a false self-loop. This is a rebuild-ordering aid, not a duplicate of `twb_calc_complexity()`: that function classifies complexity and counts dependencies, while this one gives the concrete creation sequence plus the human-readable dependency list.

Value

A tibble with columns:

build_order Integer creation sequence (NA for cyclic fields).

datasource Datasource the field belongs to.

name Human-readable field name / caption.

tableau_internal_name Bracketed internal Tableau name.

formula Raw formula string.

depends_on Comma-separated names of the calculated fields this formula directly depends on (NA when none).

n_calc_deps Integer count of direct calc-on-calc dependencies.

is_cyclic Logical; TRUE when the field is part of a dependency cycle and could not be ordered.

See Also

[twb_calc_complexity\(\)](#) for complexity classification, [twb_unused_fields\(\)](#) for calculations nothing references.

Examples

```
twb <- system.file("extdata", "rebuild_kit.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)

# "Adjusted Ratio" is built after "Profit Ratio"; the Cycle A/B pair is flagged
twb_calc_build_order(xml)

parser <- TwbParser$new(twb)
parser$get_calc_build_order()
parser$calc_build_order
```

twb_calc_complexity	<i>Classify calculated fields by complexity</i>
---------------------	---

Description

Returns every calculated field in the workbook enriched with a computation category (`calc_type`), LOD sub-type, dependency count, and dependency depth — the maximum number of calc-on-calc hops in the field's dependency chain.

Usage

```
twb_calc_complexity(x, include_parameters = FALSE)
```

Arguments

x A `TwbParser` object or an `xml2` document.

include_parameters Logical; if TRUE, include parameter fields (they always land in `calc_type = "raw"` and `dep_depth = 0`). Default FALSE.

Value

A tibble with columns:

datasource Datasource the field belongs to.

name Human-readable field name.

tableau_internal_name Bracketed internal Tableau name.

datatype Field data type.

role "measure" or "dimension".

calc_type One of "lod", "table_calc", "aggregate", "raw". Tested in that precedence order.

lod_type "fixed", "include", or "exclude"; NA if not LOD.

is_table_calc Logical; existing heuristic flag preserved for backward compatibility.

dep_depth Integer; longest chain of calc-on-calc dependencies. 0 means the field only references raw fields (or has no references).

n_deps Integer; count of distinct bracketed tokens in the formula.

formula Raw formula string.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
twb_calc_complexity(xml)
```

twb_charts	<i>Chart (mark) types per worksheet.</i>
------------	--

Description

Chart (mark) types per worksheet.

Usage

```
twb_charts(x)
```

Arguments

x TwbParser or xml2 document.

Value

Tibble with columns: worksheet, mark_types (comma-separated).

twb_colors	<i>Colors and palettes referenced in the workbook.</i>
------------	--

Description

Colors and palettes referenced in the workbook.

Usage

```
twb_colors(x)
```

Arguments

x	TwbParser or xml2 document.
---	-----------------------------

Value

Tibble with columns describing palette names and explicit colors.

twb_compatibility	<i>Report feature compatibility for migration targets</i>
-------------------	---

Description

Detects Tableau workbook features that commonly affect migrations and maps them to target-tool support levels.

Usage

```
twb_compatibility(  
  x,  
  targets = c("powerbi", "shiny", "quarto", "looker", "superset")  
)
```

Arguments

x	A TwbParser object or an xml2 document.
targets	Character vector of targets. Supported values are "powerbi", "shiny", "quarto", "looker", and "superset".

Value

A tibble with one row per detected/checked feature and target.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  twb_compatibility(parser, targets = c("powerbi", "shiny"))
}
```

twb_custom_sql

Extract Custom SQL relations from a Tableau workbook

Description

Finds every `<relation formula="...">` node that looks like a SQL statement and returns its name, type, raw SQL text, and a flag for whether it starts with SELECT or WITH.

Usage

```
twb_custom_sql(x)
```

Arguments

x A TwbParser object **or** an xml2 document.

Value

A tibble with columns:

relation_name Name attribute of the relation node.

relation_type Type attribute (e.g. "text", "table").

custom_sql Full SQL text.

is_custom_sql TRUE when the text begins with SELECT or WITH.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
twb_custom_sql(xml)
```

twb_dashboards	<i>Dashboards overview (count of zones and referenced worksheets).</i>
----------------	--

Description

Dashboards overview (count of zones and referenced worksheets).

Usage

```
twb_dashboards(x)
```

Arguments

x	TwbParser or xml2 document.
---	-----------------------------

Value

Tibble with columns: dashboard, worksheet_count, zone_count.

twb_dashboard_actions	<i>Extract dashboard and workbook actions</i>
-----------------------	---

Description

Parses <action> nodes from dashboard <actions> sections and the top-level workbook <actions> section. Returns one row per action.

Usage

```
twb_dashboard_actions(x, dashboard = NULL)
```

Arguments

x	A TwbParser object or an xml2 document.
dashboard	Optional character scalar. When supplied, only actions whose <source-sheets> include a sheet from that dashboard are returned. Pass NULL (default) to return all actions.

Value

A tibble with columns:

action_name Caption / display name of the action.

action_type "filter", "highlight", "url", "set", or another type string from the XML.

source_sheets Comma-separated list of source worksheet names.

target_sheet Target worksheet name, or NA for URL actions.

run_on Trigger: "select", "menu", or "hover".

url URL value for URL-type actions; NA otherwise.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <actions>
      <action caption="Filter 1" name="act1" type="filter">
        <source-sheets>
          <source-sheet name="Sheet1"/>
        </source-sheets>
        <target-sheets>
          <target-sheet name="Sheet2"/>
        </target-sheets>
        <run-on type="select"/>
      </action>
    </actions>
  </workbook>'
)
twb_dashboard_actions(xml)
```

twb_dashboard_filters *Filters found on dashboards and their positions.*

Description

Filters found on dashboards and their positions.

Usage

```
twb_dashboard_filters(x, dashboard = NULL)
```

Arguments

x	TwbParser or xml2 document.
dashboard	Optional dashboard name to filter to.

Value

Tibble with columns: dashboard, zone_id, zone_type, field, presentation, x, y, w, h.

twb_dashboard_layout *Full layout of dashboard zones with container hierarchy*

Description

Returns one row per zone per dashboard, including the parent-zone relationship and a tiled/floating classification.

Usage

```
twb_dashboard_layout(x, dashboard = NULL)
```

Arguments

x A TwbParser object or an xml2 document.

dashboard Optional character scalar to restrict output to one dashboard.

Value

A tibble with columns:

dashboard Dashboard name.

zone_id Zone identifier.

parent_zone_id Parent zone identifier (NA for root zones).

component_type "worksheet", "filter", "legend", "parameter_control", "text", "image", "container", or "blank".

target Referenced worksheet name or object, if applicable.

layout_type "floating" or "tiled".

x Horizontal offset, or NA.

y Vertical offset, or NA.

w Width, or NA.

h Height, or NA.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <dashboards>
      <dashboard name="Overview">
        <zones>
          <zone id="1" type="layoutV">
            <zone id="2" worksheet="Sheet1" x="0" y="0" w="600" h="400"/>
            <zone id="3" type="filter" x="0" y="400" w="600" h="50"/>
          </zone>
        </zones>
      </dashboard>
    </dashboards>
  </workbook>')
```

```

    </dashboard>
  </dashboards>
</workbook>'
)
twb_dashboard_layout(xml)

```

twb_dashboard_sheets *List worksheets embedded in each dashboard*

Description

Returns one row per worksheet per dashboard, with the zone's position on the canvas.

Usage

```
twb_dashboard_sheets(x, dashboard = NULL)
```

Arguments

x A TwbParser object or an xml2 document.
dashboard Optional character scalar to restrict output to one dashboard.

Value

A tibble with columns:

dashboard Dashboard name.
sheet Referenced worksheet name.
zone_id Zone identifier.
x Horizontal offset (pixels), or NA.
y Vertical offset (pixels), or NA.
w Width (pixels), or NA.
h Height (pixels), or NA.

Examples

```

xml <- xml2::read_xml(
  '<workbook>
    <dashboards>
      <dashboard name="Overview">
        <zones>
          <zone id="1" worksheet="Sheet1" x="0" y="0" w="600" h="400"/>
        </zones>
      </dashboard>
    </dashboards>
  </workbook>'
)
twb_dashboard_sheets(xml)

```

twb_dashboard_size *Dashboard page size and sizing mode*

Description

Reads the <size> element of each dashboard, which records how the page is sized: a fixed pixel size, an automatic (fit-to-window) size, or a range.

Usage

```
twb_dashboard_size(x, dashboard = NULL)
```

Arguments

x A TwbParser object or an xml2 document.

dashboard Optional character scalar to restrict output to one dashboard.

Value

A tibble with columns:

dashboard Dashboard name.

sizing_mode "fixed", "automatic", "range", or another mode string; NA if no <size> element is present.

min_width Minimum width in pixels, or NA.

min_height Minimum height in pixels, or NA.

max_width Maximum (or fixed) width in pixels, or NA.

max_height Maximum (or fixed) height in pixels, or NA.

Examples

```
xml <- xml2::read_xml(
  '<workbook><dashboards>
    <dashboard name="Overview">
      <size sizing-mode="fixed" maxwidth="1200" maxheight="800"
        minwidth="1200" minheight="800"/>
    </dashboard>
  </dashboards></workbook>'
)
twb_dashboard_size(xml)
```

twb_dashboard_summary	<i>Per-dashboard summary (filters count and chart types).</i>
-----------------------	---

Description

Per-dashboard summary (filters count and chart types).

Usage

```
twb_dashboard_summary(x)
```

Arguments

x	TwbParser or xml2 document.
---	-----------------------------

Value

Tibble with columns: dashboard, worksheet_count, zone_count, filters, chart_types.

twb_field_usage	<i>Field usage matrix across worksheets</i>
-----------------	---

Description

Combines shelf placement and filter usage into a tidy long tibble showing where each field appears and in what capacity across all (or selected) worksheets.

Usage

```
twb_field_usage(  
  x,  
  include_filters = TRUE,  
  include_shelves = TRUE,  
  wide = FALSE  
)
```

Arguments

x	A TwbParser object or an xml2 document.
include_filters	Logical; include filter appearances. Default TRUE.
include_shelves	Logical; include shelf appearances (rows, cols, color, size, etc.). Default TRUE.
wide	Logical; if TRUE, pivot to one row per field with one column per sheet containing a comma-separated list of contexts, or NA if the field does not appear on that sheet. Default FALSE.

Value

Long form (wide = FALSE): a tibble with columns:

field_clean Human-readable field name.

datasource Datasource the field belongs to.

sheet Worksheet name.

context Usage context, e.g. "shelf:rows", "shelf:color", "filter".

n_appearances Number of times the field appears in this context on this sheet (handles multi-pill rows/cols).

Wide form (wide = TRUE): one row per (field_clean, datasource), one column per sheet, cell value is a comma-separated context string or NA.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
twb_field_usage(xml)
twb_field_usage(xml, wide = TRUE)
```

twb_formatting	Formatting rules (fonts, colours, number formats, ...)
----------------	--

Description

Walks every <format> entry inside a <style-rule> and returns a tidy table of formatting attributes. This is where fonts, colours, number/date formats, alignment, and shading live in a Tableau workbook. Filter the attr column for what you need (e.g. attr == "number-format", or startsWith(attr, "font")).

Usage

```
twb_formatting(x, scope = NULL)
```

Arguments

- x A TwbParser object or an xml2 document.
- scope Optional character scalar ("worksheet", "dashboard", or "workbook") to restrict output.

Value

- A tibble with columns:
- scope** "worksheet", "dashboard", or "workbook".
 - container** Worksheet or dashboard name (NA for workbook scope).
 - element** The styled element, e.g. "mark", "axis", "pane", "cell", "title".
 - field** Field the rule targets, if any; NA otherwise.
 - target** Palette bucket or other target value, if any; NA otherwise.
 - attr** Formatting attribute, e.g. "font-family", "font-size", "color", "number-format", "encoding-color".
 - value** Attribute value.

Examples

```
xml <- xml2::read_xml(
  '<workbook><worksheets><worksheet name="Sales"><table><style>
    <style-rule element="mark">
      <format attr="mark-labels-cull" value="true"/>
      <format attr="number-format" value="$#,##0"/>
    </style-rule>
  </style></table></worksheet></worksheets></workbook>'
)
twb_formatting(xml)
```

twb_initial_sql	Extract Initial SQL statements from Tableau connections
-----------------	---

Description

Returns any <initial-sql> nodes found inside connection or named-connection elements.

Usage

```
twb_initial_sql(x)
```

Arguments

- x A TwbParser object **or** an xml2 document.

Value

- A tibble with columns:
- connection_id** Name or caption of the parent connection element.
 - initial_sql** SQL text of the initial statement.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
twb_initial_sql(xml)
```

twb_lineage

*Build workbook lineage for migration analysis***Description**

Produces a graph-shaped representation of workbook dependencies from datasources and tables through fields, calculated fields, worksheets, and dashboards.

Usage

```
twb_lineage(
  x,
  format = c("tables", "igraph", "mermaid"),
  include_calc_dependencies = TRUE
)
```

Arguments

x A TwbParser object or an xml2 document.

format Output format: "tables" returns list(nodes, edges); "igraph" returns an igraph object; "mermaid" returns a Mermaid flowchart string.

include_calc_dependencies Logical; include field-to-calculation and calculation-to-calculation dependencies parsed from formulas. Default TRUE.

Value

Depends on format. The default is a list with:

nodes Tibble with id, label, and type.

edges Tibble with from, to, and relationship.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  lineage <- twb_lineage(parser)
  lineage$nodes
  lineage$edges
}
```

twb_migration_assessment

Assess Tableau workbook migration readiness

Description

Scores workbook complexity and summarizes likely migration risks for a target visualization tool.

Usage

```
twb_migration_assessment(
  x,
  target = c("powerbi", "shiny", "quarto", "looker", "superset")
)
```

Arguments

x	A TwbParser object or an xml2 document.
target	Target tool. One of "powerbi", "shiny", "quarto", "looker", or "superset".

Value

A named list with summary, compatibility, and recommendations.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  twb_migration_assessment(parser, target = "shiny")$summary
}
```

twb_pages

List all pages (dashboards, worksheets, stories).

Description

List all pages (dashboards, worksheets, stories).

Usage

```
twb_pages(x)
```

Arguments

x	TwbParser or xml2 document.
---	-----------------------------

Value

Tibble with columns: page_type, name.

twb_pages_summary	<i>Summary of all pages (counts and quick descriptors).</i>
-------------------	---

Description

Summary of all pages (counts and quick descriptors).

Usage

```
twb_pages_summary(x)
```

Arguments

x	TwbParser or xml2 document.
---	-----------------------------

Value

Tibble with columns including page_type, name, and count columns such as n_zones, n_worksheets, n_filters, n_legends, n_parameter_controls, n_story_points, and mark_types for worksheets.

twb_page_composition	<i>Show what a specific page is composed of.</i>
----------------------	--

Description

For a dashboard: one row per zone with component type, target (worksheet or field), filter presentation (if applicable), and x/y/w/h when present. For a worksheet: mark types, filters, legends, parameter controls. For a story: one row per story point with its referenced target.

Usage

```
twb_page_composition(x, name)
```

Arguments

x	TwbParser or xml2 document.
name	Page name (character scalar).

Value

Tibble with columns: page_type, page_name, component_type, zone_id, target, field, presentation, x, y, w, h.

twb_parameter_usage	<i>Where each parameter is consumed</i>
---------------------	---

Description

One row per parameter usage: calculated-field formulas that reference the parameter, worksheet shelves and filters it appears on, and dashboard filter zones bound to it. Parameters become variables when rebuilding in another tool, so this is the map of everywhere each variable's value flows.

Usage

```
twb_parameter_usage(x)
```

Arguments

x	A TwbParser object or an xml2 document.
---	---

Details

Parameters with no usages do not appear here; find them with [twb_unused_fields\(\)](#). There is no overlap with [twb_field_usage\(\)](#): that function maps *fields* to worksheets, while this one maps *parameters* to every consumption point including formulas.

Value

A tibble with columns:

parameter Human-readable parameter name / caption.

datasource Datasource the parameter belongs to.

datatype Parameter data type.

current_value Current value if specified in the workbook.

context Where it is used: "formula", "shelf:<shelf>" (e.g. "shelf:rows"), "filter", or "dashboard_filter".

location Calculated field name, worksheet name, or dashboard name, depending on context.

See Also

[twb_unused_fields\(\)](#) for parameters nothing references, [extract_parameters\(\)](#) for parameter definitions.

Examples

```
twb <- system.file("extdata", "rebuild_kit.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)

# "Top N" is used in the "Adjusted Ratio" formula and on a worksheet filter
twb_parameter_usage(xml)

parser <- TwbParser$new(twb)
parser$get_parameter_usage()
parser$parameter_usage
```

twb_published_refs	<i>Detect references to published data sources</i>
--------------------	--

Description

Inspects datasource nodes and heuristically flags those that reference a published (server-side) source rather than an embedded one.

Usage

```
twb_published_refs(x)
```

Arguments

x A TwbParser object **or** an xml2 document.

Value

A tibble with columns:

name Internal datasource name.

caption User-visible caption.

hasconn Value of the hasconnection attribute.

likely_published TRUE when hasconnection = false or when the node text contains published-source markers.

hints Short explanation of the classification.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
twb_published_refs(xml)
```

twb_replication_brief *Replication brief for a Tableau workbook or dashboard*

Description

Assembles all extracted intelligence — datasources, parameters, calculated fields with complexity classifications, field usage, filters, sorts, chart types, dashboard layout, and actions — into a single named list (or formatted text) ready for use when porting to another visualisation tool.

Usage

```
twb_replication_brief(
  x,
  dashboard = NULL,
  include_sql = TRUE,
  include_formulas = TRUE,
  format = c("list", "text")
)
```

Arguments

x	A TwbParser object or an xml2 document.
dashboard	Optional character scalar. When supplied, sheet-level sections (filters, sorts, chart types, field usage, layout) are scoped to the sheets that belong to this dashboard.
include_sql	Logical; include custom SQL blocks in \$custom_sql. Default TRUE.
include_formulas	Logical; when TRUE, a formula_pretty column is added to \$calculated_fields. Default TRUE.
format	Either "list" (default) to return a named R list, or "text" to return a single formatted character string suitable for printing or writing to a file.

Value

format = "list": a named list with elements:

meta 1-row tibble: file name, counts, generation timestamp.

datasources Datasource connection details.

parameters Parameter fields with current values.

custom_sql Custom SQL blocks, or NULL if include_sql = FALSE.

calculated_fields Tibble from [twb_calc_complexity\(\)](#), optionally with a formula_pretty column.

field_usage Tibble from [twb_field_usage\(\)](#).

filters Worksheet filters (scoped to dashboard if given).

sorts Worksheet sorts (scoped to dashboard if given).

chart_types Mark types per worksheet.

dashboard_layout Zone positions from `twb_dashboard_sheets()`.

actions Dashboard actions from `twb_dashboard_actions()`.

format = "text": a single Markdown character(1) with section headers, compact tables, and fenced formula/SQL blocks.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)
brief <- twb_replication_brief(xml)
names(brief)
brief$meta
```

twb_sheet_axes	<i>Extract axis configuration for worksheets</i>
----------------	--

Description

Reads per-axis style rules embedded in worksheet XML and returns one row per axis per worksheet.

Usage

```
twb_sheet_axes(x, sheet = NULL)
```

Arguments

x	A TwbParser object or an xml2 document.
sheet	Optional character scalar to restrict output to one worksheet.

Value

A tibble with columns:

sheet Worksheet name.

axis Axis identifier (e.g., "rows", "cols", or "automatic").

field_ref Column reference if axis is field-specific; NA otherwise.

field_clean Human-readable field name; NA if not field-specific.

scale_type Scale type ("linear", "log", ...) if present; NA otherwise.

reversed Logical: TRUE if axis is reversed.

include_zero Logical: TRUE if zero is pinned on axis.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <worksheets>
      <worksheet name="Sheet1">
        <table>
          <style>
            <style-rule element="axis">
              <format attr="reverse" value="false"/>
              <format attr="scale-include-zero" value="true"/>
            </style-rule>
          </style>
        </table>
      </worksheet>
    </worksheets>
  </workbook>'
)
twb_sheet_axes(xml)
```

twb_sheet_filters

Extract detailed filter configuration for worksheets

Description

Returns one row per filter per worksheet, with full details on filter class, inclusion mode, categorical members, and numeric or date range bounds.

Usage

```
twb_sheet_filters(x, sheet = NULL)
```

Arguments

x A TwbParser object or an xml2 document.

sheet Optional character scalar to restrict output to one worksheet.

Value

A tibble with columns:

sheet Worksheet name.

field_ref Raw column-reference attribute value.

field_clean Human-readable field name.

datasource Datasource name.

filter_class Tableau filter class: "categorical", "range", "relative-date", "date", "set", "top", etc.

include_mode "include" or "exclude".

members Comma-separated member values for categorical filters; NA otherwise.

range_min Lower bound for range/quantitative filters; NA otherwise.

range_max Upper bound; NA otherwise.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <worksheets>
      <worksheet name="Sheet1">
        <table>
          <view>
            <filter class="categorical" column="[ds].[Category]">
              <groupfilter function="union">
                <groupfilter function="member" member="[Furniture]"/>
                <groupfilter function="member" member="[Technology]"/>
              </groupfilter>
            </filter>
          </view>
        </table>
      </worksheet>
    </worksheets>
  </workbook>'
)
twb_sheet_filters(xml)
```

twb_sheet_shelves	<i>Extract field-to-shelf assignments for worksheets</i>
-------------------	--

Description

Returns a tidy tibble describing which fields are placed on each visual shelf (rows, cols, color, size, label, detail, tooltip, etc.) for every worksheet in the workbook (or a single named sheet).

Usage

```
twb_sheet_shelves(x, sheet = NULL)
```

Arguments

x	A TwbParser object or an xml2 document.
sheet	Optional character scalar. When supplied only that worksheet is returned; otherwise all worksheets are returned.

Value

A tibble with columns:

sheet Worksheet name.

shelf Shelf name: "rows", "cols", or an encoding type such as "color", "size", "label", "detail", "tooltip", "shape", "text", "path", "angle", "lod", "geometry", etc.

field_ref Raw column-reference attribute value.

field_instance Field instance name (after stripping datasource prefix).

field_clean Human-readable field name.

datasource Datasource name referenced.

aggregation Aggregation function ("SUM", "AVG", ...) or NA.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <worksheets>
      <worksheet name="Sales">
        <table>
          <rows>[ds].[Category]</rows>
          <cols>[ds].[Sales]</cols>
          <panes>
            <pane>
              <mark class="Bar"/>
              <encodings>
                <color column="[ds].[Category]"/>
              </encodings>
            </pane>
          </panes>
        </table>
      </worksheet>
    </worksheets>
  </workbook>'
)
twb_sheet_shelves(xml)
```

twb_sheet_sorts

Extract sort configuration for worksheets

Description

Returns one row per sort directive per worksheet.

Usage

```
twb_sheet_sorts(x, sheet = NULL)
```

Arguments

<code>x</code>	A TwbParser object or an xml2 document.
<code>sheet</code>	Optional character scalar to restrict output to one worksheet.

Value

A tibble with columns:

sheet Worksheet name.

field_ref Raw column-reference attribute.

field_clean Human-readable field name.

datasource Datasource name.

sort_order "ascending" or "descending".

sort_by Sort method: "field", "alphabetic", "manual", "data-source-order", etc.

Examples

```
xml <- xml2::read_xml(
  '<workbook>
    <worksheets>
      <worksheet name="Sheet1">
        <table>
          <view>
            <sort class="sum" column="[ds].[Sales]" direction="descending"/>
          </view>
        </table>
      </worksheet>
    </worksheets>
  </workbook>'
)
twb_sheet_sorts(xml)
```

twb_sheet_spec

Worksheet visualization specs and dashboard chart inventories

Description

`twb_sheet_spec()` reduces a worksheet to everything needed to understand — and rebuild — its visualization in another tool: the mark type, the fields on the rows and columns shelves (in order), the dimensions and measures in play, every marks-card encoding (color, size, label, detail, shape, tooltip, ...), tooltip configuration, and the worksheet's filters, sorts, and axes.

`twb_dashboard_charts()` answers "what graphs are on this dashboard?": one row per worksheet placed on each dashboard, with its mark type, fields, tooltip summary, and layout position.

Usage

```
twb_sheet_spec(x, sheet = NULL)
```

```
twb_dashboard_charts(x, dashboard = NULL)
```

Arguments

<code>x</code>	A TwbParser object or an xml2 document.
<code>sheet</code>	Optional character scalar to restrict <code>twb_sheet_spec()</code> to one worksheet.
<code>dashboard</code>	Optional character scalar to restrict <code>twb_dashboard_charts()</code> to one dashboard.

Value

`twb_sheet_spec()` returns a named list (one element per worksheet) of class `twb_sheet_spec`. Each spec is a list with:

`sheet` Worksheet name.

`mark_type` Detected mark type, e.g. "bar", "line", "map", "text". "automatic" when the workbook leaves it to Tableau.

`mark_source` "explicit" when the workbook names the mark type, "inferred" for "automatic".

`datasources` Character vector of datasources referenced.

`rows, cols` Character vectors of clean field names on the rows and columns shelves, in shelf order.

`dimensions, measures` Character vectors of the dimension and measure fields used anywhere in the visualization. A pill counts as a measure when it is aggregated; otherwise its declared field role wins.

`encodings` Tibble with channel, field, aggregation, `used_as` ("dimension"/"measure") for every marks-card encoding.

`shelves` Tibble with shelf ("rows"/"cols"), field, aggregation, `used_as` for the row/column pills, in shelf order.

`tooltip` List with `has_tooltip`, `customized`, `text`, and `fields`.

`filters, sorts, axes` Tibbles from `twb_sheet_filters()`, `twb_sheet_sorts()`, and `twb_sheet_axes()`.

`twb_dashboard_charts()` returns a tibble with one row per worksheet placed on a dashboard: `dashboard`, `sheet`, `mark_type`, `mark_source`, `rows`, `cols`, `dimensions`, `measures`, `tooltip_fields` (list-columns), `n_tooltip_fields`, `has_tooltip`, `n_filters`, `datasources`, and the layout `zone_id`, `x`, `y`, `w`, `h`.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  spec <- twb_sheet_spec(parser)
  spec
```

```

    twb_dashboard_charts(parser)
  }

```

twb_tooltips

Worksheet tooltips

Description

Extracts the (plain-text) tooltip configured for each worksheet. Tableau stores customised tooltips as formatted-text runs; this returns the concatenated text with markup stripped.

Usage

```
twb_tooltips(x, sheet = NULL)
```

Arguments

x	A TwbParser object or an xml2 document.
sheet	Optional character scalar to restrict output to one worksheet.

Value

A tibble with columns:

sheet Worksheet name.

is_customized TRUE if a non-empty customised tooltip is present.

tooltip_text Plain-text tooltip content, or NA.

Examples

```

xml <- xml2::read_xml(
  '<workbook><worksheets><worksheet name="Sales">
    <customized-tooltip><formatted-text>
      <run>Sales: </run><run>&lt;SUM(Sales)&gt;</run>
    </formatted-text></customized-tooltip>
  </worksheet></worksheets></workbook>'
)
twb_tooltips(xml)

```

twb_unused_fields	<i>Fields defined in the workbook but never used</i>
-------------------	--

Description

Lists every raw field, calculated field, and parameter that is defined in a datasource but referenced nowhere: not on any worksheet shelf, filter, or sort, not in any tooltip, not inside any other calculated-field formula, and not in any dashboard filter zone. When rebuilding a workbook in another tool, these are the fields that can safely be left behind.

Usage

```
twb_unused_fields(x)
```

Arguments

x A TwbParser object or an xml2 document.

Details

Matching is deliberately conservative: a field is reported as unused only when none of its name forms (internal name or display caption, in any bracket/derivation wrapping) appears anywhere. A field whose name merely collides with a used field in another datasource is treated as used.

This complements [twb_field_usage\(\)](#), which shows where each *used* field appears across worksheets.

Value

A tibble with columns:

datasource Datasource the field belongs to.

field_type "raw", "calculated", or "parameter".

name Human-readable field name / caption.

tableau_internal_name Bracketed internal Tableau name.

datatype Field data type.

role "measure" or "dimension".

is_hidden Whether a raw field is hidden (NA for calculated fields and parameters).

See Also

[twb_field_usage\(\)](#) for where used fields appear, [twb_calc_build_order\(\)](#) for rebuilding calculations in dependency order.

Examples

```
twb <- system.file("extdata", "rebuild_kit.twb", package = "twbparser")
stopifnot(nzchar(twb), file.exists(twb))
xml <- xml2::read_xml(twb)

# the fixture leaves exactly one raw field, one calc, and one parameter unused
twb_unused_fields(xml)

# same result through the parser object
parser <- TwbParser$new(twb)
parser$get_unused_fields()
parser$unused_fields
```

 validate_relationships

Validate relationships against available datasources and fields

Description

Checks that relationship endpoints reference known datasource tables and that the predicate fields appear somewhere in the workbook (calculated, raw, or parameter fields), using a lenient token match (e.g., INT([GEOID]) = GEOID).

Usage

```
validate_relationships(parser, strict = FALSE)
```

Arguments

parser	A TwbParser-like object that exposes: get_relationships(), get_datasources(), get_fields(), and get_calculated_fields(). (S3/R6 both fine.)
strict	Deprecated and ignored. Kept for backward compatibility with twbparser 0.5.0, where it was reserved for future table-scoped checks.

Value

A list with:

ok TRUE if no issues; FALSE otherwise.

issues A named list of tibbles. Possible elements:

- unknown_tables: endpoints not found among known tables.
- unknown_fields: predicate fields not found in the field pool.

Examples

```
twb <- system.file("extdata", "test_for_wenjie.twb", package = "twbparser")
if (nzchar(twb) && file.exists(twb)) {
  parser <- TwbParser$new(twb)
  res <- validate_relationships(parser)
  if (!res$ok) print(res$issues)
}
```

Index

audit_tableau_folder, 3

build_dependency_graph, 4
build_dependency_graph(), 17

export_migration_bundle, 5
extract_calculated_fields, 6
extract_columns_with_table_source, 7
extract_columns_with_table_source(), 16
extract_datasource_details, 8
extract_joins, 9
extract_named_connections, 10
extract_parameters, 11
extract_parameters(), 8, 44
extract_raw_fields, 12
extract_relations, 13
extract_relationships, 14
extract_twb_from_twbx, 15

infer_implicit_relationships, 15

parse_twb, 16
plot_dependency_graph, 17
plot_relationship_graph, 18
plot_source_join_graph, 18
prettify_calculated_fields, 19

render_migration_brief, 20
run_twbparser_app, 20

scaffold_quarto_dashboard, 21
scaffold_shiny_dashboard, 21
shiny::runApp(), 20

tableau_formula_pretty, 22
translate_tableau_calc, 22
twb_calc_build_order, 28
twb_calc_build_order(), 54
twb_calc_complexity, 29
twb_calc_complexity(), 28, 29, 46

twb_charts, 30
twb_colors, 31
twb_compatibility, 31
twb_custom_sql, 32
twb_dashboard_actions, 33
twb_dashboard_actions(), 47
twb_dashboard_charts(twb_sheet_spec), 51
twb_dashboard_filters, 34
twb_dashboard_layout, 35
twb_dashboard_sheets, 36
twb_dashboard_sheets(), 47
twb_dashboard_size, 37
twb_dashboard_summary, 38
twb_dashboards, 33
twb_field_usage, 38
twb_field_usage(), 44, 46, 54
twb_formatting, 39
twb_initial_sql, 40
twb_lineage, 41
twb_migration_assessment, 42
twb_page_composition, 43
twb_pages, 42
twb_pages_summary, 43
twb_parameter_usage, 44
twb_published_refs, 45
twb_replication_brief, 46
twb_sheet_axes, 47
twb_sheet_axes(), 52
twb_sheet_filters, 48
twb_sheet_filters(), 52
twb_sheet_shelves, 49
twb_sheet_sorts, 50
twb_sheet_sorts(), 52
twb_sheet_spec, 51
twb_tooltips, 53
twb_unused_fields, 54
twb_unused_fields(), 29, 44
TWBParser (TwbParser), 23

TwbParser, [16](#), [23](#)
twbx_extract_files, [27](#)
twbx_list, [27](#)
twbx_list(), [27](#)
validate_relationships, [55](#)