Package: twbparser (via r-universe)

May 11, 2026

Title Parse 'Tableau' Workbooks into Functional Data

Version 0.4.0

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.

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, spelling, styler, testthat (>= 3.0.0), tidygraph, vctrs, zip

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData false

Roxygen list(markdown = TRUE, r6 = FALSE)

RoxygenNote 7.3.3

Config/pak/sysreqs libglpk-dev libicu-dev libxml2-dev

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

Date/Publication 2026-04-11 14:54:58 UTC

RemoteUrl https://github.com/prigasg/twbparser

RemoteRef HEAD

RemoteSha 61768036672d96811ee38d2a18209e299b226240

Contents

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)
```

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 a filtered table of parameter datasources.

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 datasources (if present).

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` *Extract Tableau join clauses from* `<relation type="join">` *nodes*

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` *Extract parameter fields from a TWB*

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` *Extract non-calculated, non-parameter fields from a TWB*

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` *Extract all* `<relation>` *tags from a TWB*

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:

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

---

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`

`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)

`tbs_custom_sql_graphql` *Custom SQL (Metadata API) for a published item*

Description

Queries the Metadata (GraphQL) API for Custom SQL tables in the content graph.

Usage

```
tbs_custom_sql_graphql(
  content_id,
  base_url = Sys.getenv("TABLEAU_BASE_URL"),
  site = Sys.getenv("TABLEAU_SITE"),
  token = Sys.getenv("TABLEAU_PAT")
)
```

Arguments

content_id	Character. Workbook or datasource ID (GUID).
base_url	Character. Server/Cloud base URL (e.g., "https://...").
site	Character. Site contentUrl ("" for default site).
token	Character. REST credentials token.

Value

A tibble with columns such as custom_sql_name, custom_sql_query, database, schema. Zero rows if none.

Examples

```
tbs_custom_sql_graphql("abc-123")
```

---

tbs_publish_info *Publish info for a workbook or datasource on 'Tableau' Server/Cloud*

Description

Returns an empty tibble when credentials are missing or the item is not found.

Usage

```
tbs_publish_info(
  content_id,
  base_url = Sys.getenv("TABLEAU_BASE_URL"),
  site = Sys.getenv("TABLEAU_SITE"),
  token = Sys.getenv("TABLEAU_PAT")
)
```

Arguments

content_id	Character. Workbook or datasource ID (GUID).
base_url	Character. Server/Cloud base URL (e.g., "https://...").
site	Character. Site contentUrl ("" for the default site).
token	Character. REST credentials token (from a prior sign-in).

Value

A tibble with columns like content_id, site, project, web_url, created_at, updated_at. May be zero rows if unavailable.

Examples

```
tbs_publish_info("abc-123")
```

`twb_calc_complexity` *Classify calculated fields by complexity*

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 *Chart (mark) types per worksheet.*

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 *Colors and palettes referenced in the workbook.*

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_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_dashboard_actions *Extract dashboard and workbook actions*

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>'
)
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_summary *Per-dashboard summary (filters count and chart types).*

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_dashboards *Dashboards overview (count of zones and referenced worksheets).*

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_field_usage *Field usage matrix across worksheets*

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_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_page_composition *Show what a specific page is composed of.*

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_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 *Summary of all pages (counts and quick descriptors).*

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_published_refs *Detect references to published data sources*

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 character(1) with section headers and tabular output.

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 *Extract axis configuration for worksheets*

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` *Extract field-to-shelf assignments for worksheets*

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

`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.

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)
```

`TwbParser` *Tableau Workbook Parser (R6)*

Description

Initialize the parser from a `.twb` or `.twbx` path.

Return the TWBX manifest (if available).

Return TWBX extract entries.

Return TWBX image entries.

Extract files from the TWBX to disk.

Fields placed on visual shelves for one or all worksheets.

Detailed filter configuration for one or all worksheets.

Axis configuration for one or all worksheets.

Sort directives for one or all worksheets.

Worksheets embedded in one or all dashboards.

Full zone layout with container hierarchy.

Dashboard and workbook actions.

Calculated field complexity classifications.

Field usage matrix across worksheets.

Full replication brief for the workbook or a single dashboard.

Validate relationships; optionally stop on failure.

Print a concise summary of parsed content.

Arguments

`path`	Path to a `.twb` or `.twbx` file.
`types`	Optional vector of types (e.g., `"image"`, `"extract"`).
`pattern`	Optional regex to match archive paths.
`files`	Optional explicit archive paths to extract.
`exdir`	Output directory (defaults to parser's twbx dir or tempdir()).
`sheet`	Optional worksheet name.
`include_parameters`	Logical; include parameter fields. Default `FALSE`.
`include_filters`	Include filter appearances. Default `TRUE`.
`include_shelves`	Include shelf appearances. Default `TRUE`.
`wide`	Return wide format (one col per sheet). Default `FALSE`.
`dashboard`	Optional dashboard name to scope the brief.

include_sql	Include custom SQL blocks. Default TRUE.
include_formulas	Add formula_pretty to calculated fields. Default TRUE.
format	"list" (default) or "text".
error	If TRUE, stop() when validation fails.

Format

An R6 class generator.

Details

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.

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.

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

Methods

new(path) Create a parser from .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, pattern, files, exdir) Extract files from .twbx archive.

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) Return calculated fields tibble. When `pretty = TRUE`, includes a `formula_pretty` column with line breaks and indentation.

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

summary() Print a brief summary to console.

`twbx_extract_files` *Extract specific files from a .twbx*

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 *List contents of a Tableau .twbx*

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)
```

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	Logical. Reserved for future table-scoped checks that can be overly conservative with federated sources. Currently not used.

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