Package: ecospgr (via r-universe)

June 10, 2026

Title Shared Loader Utilities for ecospg

Version 0.3.0

Description Extracts duplicated R code from ecospg data loaders into a reusable package. Provides database connection, catalog registration, partitioned table management, staging/ingest, S3 upload, and CLI argument parsing utilities.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports DBI, RPostgres, glue, cli, jsonlite, fs, sf

Suggests testthat (>= 3.0.0), withr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libuv1-dev libssl-dev libpq-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-06-10 10:55:00 UTC

RemoteUrl https://gitlab.au.dk/ecos/tools/r-pkgs/ecospgr

RemoteRef HEAD

RemoteSha 365a1accfdbbbbf36f87409d920fc5a8bf8be173

Contents

`epg_build_source_meta` *Build base source metadata*

Description

Build base source metadata

Usage

```
epg_build_source_meta(source_url, metadata_url = NULL, extra = list())
```

Arguments

`source_url`	The source URL for the download.
`metadata_url`	Optional metadata URL.
`extra`	Named list of additional metadata fields to merge.

Value

A named list of source metadata.

epg_collect_source_meta_overrides

Collect source metadata overrides from file and/or inline JSON

Description

Reads optional JSON from a file path and/or an inline JSON string and merges them (inline takes precedence over file).

Usage

```
epg_collect_source_meta_overrides(json_file_path = NULL, json_inline = NULL)
```

Arguments

`json_file_path`	Path to a JSON file (or NULL).
`json_inline`	Inline JSON string or `@file` reference (or NULL).

Value

A named list of override fields (may be empty).

epg_create_partition

Create a partition table and indexes for a dataset version

Description

Uses `glue_sql()` for safe SQL interpolation. Identifier components are validated to contain only safe characters (alphanumeric + underscore).

Usage

```
epg_create_partition(con, dataset, ver_id, geom_type = "MultiPolygon")
```

Arguments

`con`	A DBI connection.
`dataset`	Dataset name (used as part of table names in `vector` schema).
`ver_id`	The dataset_version.id (integer).
`geom_type`	Geometry type constraint for the partition. Defaults to `"MultiPolygon"`. Use `"Geometry"` for broader geometry support (e.g., geodk datasets).

epg_detect_xsd_for_gml *Detect a companion XSD file for a GML file*

Description

Looks for schemaLocation hints in the GML header, then falls back to filename-based matching and sibling XSD files.

Usage

```
epg_detect_xsd_for_gml(gml_path)
```

Arguments

gml_path	Path to a GML file.

Value

Path to the XSD file, or NA_character_ if none found.

epg_dfd_auth_params *Datafordeler authentication query parameters*

Description

Returns a named list of credential query parameters for Datafordeler URLs. Prefers API-key auth (apikey=) when DFD_API_KEY is set; falls back to legacy username/password (username=. . . &password=. . .). Errors if neither is available.

Usage

```
epg_dfd_auth_params(
  api_key = Sys.getenv("DFD_API_KEY", ""),
  username = Sys.getenv("DFD_USERNAME", ""),
  password = Sys.getenv("DFD_PASSWORD", "")
)
```

Arguments

api_key	API key (default Sys.getenv("DFD_API_KEY"))
username	Legacy service-user name (default Sys.getenv("DFD_USERNAME"))
password	Legacy service-user password (default Sys.getenv("DFD_PASSWORD"))

Value

Named list with either apikey OR username + password.

epg_dfd_auth_qs *Datafordeler authentication query string*

Description

Returns a URL-encoded query-string fragment for the credentials returned by `epg_dfd_auth_params()`, e.g. `"apikey=abc123"` or `"username=u&password=p"`. Does not include a leading `?` or `&`.

Usage

```
epg_dfd_auth_qs(...)
```

Arguments

`...`	Passed to `epg_dfd_auth_params()`.

Value

Character scalar.

epg_ensure_dataset *Ensure a dataset row exists in meta.dataset*

Description

Ensure a dataset row exists in meta.dataset

Usage

```
epg_ensure_dataset(con, topic, dataset)
```

Arguments

`con`	A DBI connection.
`topic`	Topic name (must already exist).
`dataset`	Dataset name.

epg_ensure_topic *Ensure a topic row exists in meta.topic*

Description

Ensure a topic row exists in meta.topic

Usage

```
epg_ensure_topic(con, topic, description = NULL)
```

Arguments

con	A DBI connection.
topic	Topic name.
description	Optional topic description.

epg_finalize_version *Finalize a dataset version: mark latest and drop staging*

Description

Finalize a dataset version: mark latest and drop staging

Usage

```
epg_finalize_version(
  con,
  topic,
  dataset,
  version_tag,
  stage_tbl,
  set_latest = TRUE
)
```

Arguments

con	A DBI connection.
topic	Topic name.
dataset	Dataset name.
version_tag	The version tag string.
stage_tbl	The qualified staging table name to drop.
set_latest	Whether to mark this version as latest. Default TRUE.

epg_insert_normalized *Insert normalized geometries from staging into a vector partition*

Description

Insert normalized geometries from staging into a vector partition

Usage

```
epg_insert_normalized(
  con,
  dataset,
  ver_id,
  stage_tbl,
  geom_transform =
   "ST_Multi(ST_CollectionExtract(ST_MakeValid(s.geom), 3))::geometry(MultiPolygon,25832)",
  delete_before_insert = FALSE
)
```

Arguments

con	A DBI connection.
dataset	Dataset name (parent table is vector.<dataset>).
ver_id	The dataset_version.id.
stage_tbl	The qualified staging table name (e.g., "stage.foo_s_42").
geom_transform	SQL expression for geometry normalization. Defaults to the standard MultiPolygon transform. Use "ST_Force2D(ST_MakeValid(s.geom))::geometry(Geometry,25832)" for geodk.
delete_before_insert	If TRUE, deletes existing rows for this ver_id before inserting (for idempotent re-loads like LBST). Default FALSE.

Value

The number of rows inserted (invisible).

epg_parse_arg *Parse a command-line flag argument*

Description

Parse a command-line flag argument

Usage

```
epg_parse_arg(flag, default = NULL, args = commandArgs(trailingOnly = TRUE))
```

Arguments

flag	The flag string (e.g., "–src-url").
default	Default value if flag is not found.
args	Character vector of arguments. Defaults to commandArgs(trailingOnly = TRUE).

Value

The value following the flag, or default.

---

epg_parse_json_arg *Parse a JSON value from a string or @file reference*

Description

If value starts with @, the remainder is treated as a file path whose contents are read and parsed as JSON.

Usage

```
epg_parse_json_arg(value, label)
```

Arguments

value	A JSON string or @path/to/file.json.
label	A human-readable label used in error messages.

Value

Parsed JSON as an R list, or NULL if value is empty/NULL.

epg_parse_json_payload *Parse a JSON string*

Description

Parse a JSON string

Usage

```
epg_parse_json_payload(payload, label)
```

Arguments

payload	A JSON string.
label	A human-readable label used in error messages.

Value

Parsed JSON as an R list.

epg_persist_source_meta *Persist source metadata on a dataset version*

Description

Persist source metadata on a dataset version

Usage

```
epg_persist_source_meta(con, ver_id, source_meta)
```

Arguments

con	A DBI connection.
ver_id	The dataset_version.id.
source_meta	A named list of metadata to store as JSONB.

epg_pg_connect *Connect to PostgreSQL using PG* environment variables*

Description

Reads PGHOST, PGPORT, PGDATABASE, PGUSER, PGPASSWORD from the environment with sensible defaults for local Docker development.

Usage

```
epg_pg_connect(application_name = NULL)
```

Arguments

application_name
: Optional application name tag for the connection (visible in pg_stat_activity).

Value

A PqConnection object.

epg_read_json_file *Read and parse a JSON file*

Description

Read and parse a JSON file

Usage

```
epg_read_json_file(path, label)
```

Arguments

path	Path to a JSON file.
label	A human-readable label used in error messages.

Value

Parsed JSON as an R list, or NULL if path is empty/NULL.

epg_register_version *Register a new dataset version*

Description

Register a new dataset version

Usage

```
epg_register_version(
  con,
  topic,
  dataset,
  version_tag,
  on_conflict = c("error", "reuse")
)
```

Arguments

con	A DBI connection.
topic	Topic name.
dataset	Dataset name.
version_tag	Version tag string.
on_conflict	"error" (default) to fail on duplicate, or "reuse" to return the existing version id.

Value

The dataset_version.id (integer).

epg_resolve_s3_config *Resolve S3-compatible object storage configuration from environment*

Description

Reads RAW_* env vars, falling back to SCW_* for backward compatibility.

Usage

```
epg_resolve_s3_config(default_prefix = "")
```

Arguments

default_prefix	Default S3 key prefix if RAW_PREFIX / SCW_RAW_PREFIX is not set (e.g., "raw/dataset/version").

Value

A named list with components `endpoint`, `region`, `bucket`, `prefix`, `access_key`, `secret_key`, and `configured` (logical).

`epg_run_pipeline` *Run the full ingest pipeline in a single transaction*

Description

Convenience wrapper that calls `epg_ensure_topic()`, `epg_ensure_dataset()`, `epg_register_version()`, `epg_set_raw_file_url()`, `epg_persist_source_meta()`, `epg_create_partition()`, `epg_write_staging()`, `epg_insert_normalized()`, and `epg_finalize_version()` inside `dbWithTransaction()`.

Usage

```
epg_run_pipeline(
  con,
  sf_obj,
  topic,
  dataset,
  version_tag,
  topic_description = NULL,
  source_meta = NULL,
  raw_url = NA_character_,
  geom_type = "MultiPolygon",
  geom_transform =
   "ST_Multi(ST_CollectionExtract(ST_MakeValid(s.geom), 3))::geometry(MultiPolygon,25832)",
  delete_before_insert = FALSE,
  set_latest = TRUE,
  on_conflict_version = "error"
)
```

Arguments

`con`	A DBI connection.
`sf_obj`	An sf object with the features to load.
`topic`	Topic name.
`dataset`	Dataset name.
`version_tag`	Version tag string.
`topic_description`	Optional topic description.
`source_meta`	Named list of source metadata (or NULL).
`raw_url`	Optional S3 URI of the raw file.
`geom_type`	Geometry type for the partition. Default `"MultiPolygon"`.

`geom_transform` SQL expression for geometry normalization.

`delete_before_insert`
If `TRUE`, deletes existing rows for this version before inserting.

`set_latest` Whether to mark this version as latest.

`on_conflict_version`
Passed as `on_conflict` to `epg_register_version()`.

Value

The `dataset_version.id` (invisible).

`epg_sanitize_name` *Sanitize a string into a valid SQL/table name*

Description

Lowercases, replaces non-alphanumeric characters with underscores, and strips leading/trailing underscores.

Usage

```
epg_sanitize_name(x)
```

Arguments

`x` A character string.

Value

A sanitized character string.

`epg_set_raw_file_url` *Set the raw file URL on a dataset version*

Description

Set the raw file URL on a dataset version

Usage

```
epg_set_raw_file_url(con, ver_id, raw_url)
```

Arguments

`con` A DBI connection.

`ver_id` The dataset_version.id.

`raw_url` The S3 URI of the raw file.

epg_upload_raw_files *Upload raw files to S3-compatible object storage*

Description

Upload raw files to S3-compatible object storage

Usage

```
epg_upload_raw_files(files, s3_config, on_error = c("stop", "warn"))
```

Arguments

files	Character vector of local file paths to upload.
s3_config	A list as returned by epg_resolve_s3_config().
on_error	"stop" (default) to abort on upload failure, or "warn" to emit a warning and continue.

Value

The S3 URI(s) of uploaded files (character vector), or NA_character_ for files that failed when on_error = "warn".

epg_write_staging *Write an sf object to a staging table*

Description

Creates the stage schema if needed, writes the sf object via st_write, verifies creation, normalizes the geometry column name to geom, and returns a row count.

Usage

```
epg_write_staging(con, sf_obj, dataset, ver_id)
```

Arguments

con	A DBI connection.
sf_obj	An sf object to stage.
dataset	Dataset name.
ver_id	The dataset_version.id (integer).

Value

A list with stage_tbl (qualified name like "stage.foo_s_42") and n_rows (count of rows with non-NULL geometry).

Index