Package 'fury'

January 18, 2026

Type Package

Title Pre-Analysis Data Ingestion and Audit Layer for Consumer Psychology Research

Version 0.1.0

Description Consumes validated niche_spec and niche_recipe objects to orchestrate data ingestion, produce audit artifacts, and return niche_result objects. Does not perform modeling, scoring, construct validation, or APA rendering. Part of the 'niche' R universe for reproducible consumer psychology research.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports nicheCore, vision, cli, fs, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Josh Gonzales [aut, cre] (ORCID: <https://orcid.org/0000-0001-8633-3380>)

Maintainer Josh Gonzales <jgonza10@uoguelph.ca>

Contents

fury_execute_recipe *Execute a Recipe to Produce Audit Artifacts*

Description

Takes a validated `niche_recipe` and produces audit artifacts in the specified output directory.

Usage

```
fury_execute_recipe(recipe, out_dir = tempdir())
```

Arguments

`recipe`	A `niche_recipe` object built by `vision::build_recipe()`.
`out_dir`	Character scalar. Output directory for audit artifacts. Defaults to `tempdir()`. Directory will be created if it does not exist.

Details

This is the core execution function of fury. It creates output directories, generates audit artifacts (source manifest, import log, codebook, session info), and returns a `niche_result` object.

Value

A `niche_result` object (defined by nicheCore) containing paths to audit artifacts and execution metadata.

Examples

```
## Not run:
# Assuming you have a recipe
spec <- vision::read_spec("path/to/spec.yaml")
recipe <- vision::build_recipe(spec)
result <- fury_execute_recipe(recipe)

## End(Not run)
```

fury_run *Run the Complete fury Workflow from a Spec File*

Description

Novice-friendly entry point that reads a spec file, validates it, builds a recipe, and executes data ingestion with audit artifacts.

Usage

```
fury_run(spec_path, out_dir = tempdir())
```

Arguments

spec_path	Character scalar. Path to a spec file (YAML or JSON).
out_dir	Character scalar. Output directory for audit artifacts. Defaults to tempdir(). Directory will be created if it does not exist.

Details

This function orchestrates the full fury workflow:

1. Reads and validates the spec via vision::read_spec() and vision::validate_spec()
2. Builds a recipe via vision::build_recipe()
3. Executes the recipe via fury_execute_recipe()

Value

A niche_result object (defined by nicheCore) containing audit artifacts and metadata.

Examples

```
## Not run:
# Create a minimal spec
spec_path <- tempfile(fileext = ".yaml")
vision::write_spec_template(spec_path)

# Run fury workflow
result <- fury_run(spec_path)
print(result)

## End(Not run)
```

---

fury_scope *Return the Scope Statement for fury*

Description

Returns a character string describing what fury does and does not do. Used in documentation and tests to prevent scope drift.

Usage

```
fury_scope()
```

Value

A character string describing fury's scope.

Examples

```
fury_scope()
```

`fury_write_bundle` *Write an Audit Bundle from a Result Object*

Description

Takes a `niche_result` and ensures all audit artifacts are written to the specified output directory. This is typically called automatically by `fury_execute_recipe()`, but can be used independently to re-export a bundle.

Usage

```
fury_write_bundle(result, out_dir = NULL)
```

Arguments

`result`	A `niche_result` object produced by `fury_execute_recipe()`.
`out_dir`	Character scalar. Output directory for the audit bundle. If not provided, uses the directory from the result's metadata.

Value

Invisibly returns the path to the audit bundle directory.

Examples

```
## Not run:
result <- fury_run("spec.yaml")
fury_write_bundle(result, out_dir = "my_audit")

## End(Not run)
```