Package 'scPassport'

November 12, 2026

Title Passport System for Single-Cell Objects

Version 1.0.0

Description Stamps Seurat, SingleCellExperiment, and SummarizedExperiment objects with a persistent metadata passport. For Seurat objects the passport is stored in the misc slot; for SingleCellExperiment and SummarizedExperiment objects it is stored in the metadata slot. Tracks animal info, experiment details, lineage (parent/child relationships), RDS registry numbers, processing logs, and custom fields. Includes an interactive Shiny gadget to fill and update the passport, and a read mode to print the full passport to console. The passport persists inside the RDS file with no external files needed.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 4.3)

Imports shiny, miniUI, Rcpp, S4Vectors

LinkingTo Rcpp

Suggests Seurat, SingleCellExperiment, SummarizedExperiment, knitr, rmarkdown, testthat (>= 3.0.0)

biocViews SingleCell, DataImport, Visualization, Infrastructure

VignetteBuilder knitr

Config/testthat/edition 3

BugReports https://github.com/sedatkacar56/scPassport/issues

URL https://github.com/sedatkacar56/scPassport

git_url https://git.bioconductor.org/packages/scPassport

git_branch RELEASE_3_23

git_last_commit bc4afe6

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Sedat Kacar [aut, cre] (ORCID: <https://orcid.org/0000-0002-0671-8529>)

Maintainer Sedat Kacar <sedatkacar56@gmail.com>

Contents

append_log_entry 2
build_log_entry 3
build_passport 3
log_step 5
read_passport 6
scPassport 7
validate_passport 9

Index **11**

`append_log_entry` *Append a log entry to an existing processing log*

Description

Append a log entry to an existing processing log

Usage

```
append_log_entry(log, new_entry)
```

Arguments

`log`	Existing processing log (a list of entries, may be empty).
`new_entry`	A single log entry built by `build_log_entry`.

Value

The updated log list with `new_entry` appended.

build_log_entry *Build a single processing log entry*

Description

Build a single processing log entry

Usage

```
build_log_entry(step, n_cells, n_genes, params)
```

Arguments

step	Name of the processing step
n_cells	Number of cells in the object at this step
n_genes	Number of genes in the object at this step
params	Named list of parameters used in this step

Value

A named list representing one log entry

build_passport *Build a passport list from components*

Description

Build a passport list from components

Usage

```
build_passport(
  object_id,
  rds_self,
  animal_id,
  species,
  sex,
  age,
  condition,
  tissue,
  project,
  researcher,
  date,
  notes,
  parent_id,
```

```
  rds_parent,
  lineage,
  children,
  rds_children,
  custom_keys,
  custom_vals
)
```

Arguments

object_id	Name/ID of this Seurat object
rds_self	RDS registry number of this object
animal_id	Individual animal identifier
species	Species name
sex	Sex of the animal
age	Age of the animal
condition	Experimental condition
tissue	Tissue of origin
project	Project name
researcher	Name of the researcher
date	Date of the experiment
notes	Free-text notes
parent_id	Object ID of the direct parent
rds_parent	RDS number of the parent object
lineage	Full ancestry chain as character vector
children	Object IDs of children
rds_children	RDS numbers of children objects
custom_keys	Names of custom fields
custom_vals	Values of custom fields

Value

A named list representing the passport

log_step *Log a Processing Step to a Seurat, SingleCellExperiment, or SummarizedExperiment Object*

Description

Appends a processing step entry to the object's processing log. For Seurat objects the log is stored in `@misc$processing_log`; for `SingleCellExperiment` and `SummarizedExperiment` objects it is stored in `metadata(obj)$processing_log`. Each entry records the step name, timestamp, cell count, gene count, and any parameters you pass.

Usage

```
log_step(obj, step, params = list())
```

Arguments

`obj`	A Seurat, SingleCellExperiment, or SummarizedExperiment object.
`step`	A character string describing the processing step (e.g. `"NormalizeData"`, `"ScaleData"`, `"Subset cluster 1"`).
`params`	A named list of parameters used in this step. Default is an empty list.

Value

The same object with the new log entry appended to its processing log.

See Also

scPassport, read_passport

Examples

```
# Log a step on a minimal SummarizedExperiment object
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    se <- SummarizedExperiment::SummarizedExperiment()
    se <- log_step(se, "example step")
}


# Log steps on a Seurat object (requires an existing Seurat object 'seu')
# seu <- log_step(seu, "QC filter",
#         params = list(min_cells = 3, min_features = 200))
#
# seu <- NormalizeData(seu)
# seu <- log_step(seu, "NormalizeData",
#         params = list(method = "LogNormalize", scale_factor = 10000))
#
# seu <- RunPCA(seu)
```

```
# seu <- log_step(seu, "RunPCA", params = list(npcs = 30))
#
# read_passport(seu)
```

`read_passport` *Read the Passport of a Seurat, SingleCellExperiment, or SummarizedExperiment Object*

Description

Prints the full passport stored in the object's passport slot to the console, including all known fields, any custom fields, and the processing log. Works with Seurat (`@misc$passport`), SingleCellExperiment, and SummarizedExperiment (`metadata(obj)$passport`) objects.

Usage

```
read_passport(obj)
```

Arguments

`obj`	A Seurat, SingleCellExperiment, or SummarizedExperiment object with a passport stamped via `scPassport`.

Value

Invisibly returns `NULL`. Output is printed to console.

See Also

`scPassport`, `log_step`

Examples

```
# Read passport on an unstamped object (prints "No passport found")
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    se <- SummarizedExperiment::SummarizedExperiment()
    read_passport(se)
}


# Read passport on a stamped Seurat object (requires existing object 'WTHeme')
# read_passport(WTHeme)
#
# Or via scPassport with read = TRUE:
# scPassport(WTHeme, read = TRUE)
```

scPassport *scPassport: Passport System for Single-Cell Objects*

Description

Stamps Seurat, SingleCellExperiment, and SummarizedExperiment objects with a persistent metadata passport. For Seurat objects the passport is stored in `@misc$passport`; for SingleCellExperiment and SummarizedExperiment objects it is stored in `metadata(obj)$passport`. Tracks animal info, experiment details, lineage (parent/child relationships), RDS registry numbers, processing logs, and custom fields. Includes an interactive Shiny popup to fill and update the passport, and a read mode to print it to console.

Opens an interactive Shiny popup that allows you to fill in metadata (animal info, experiment details, lineage, RDS numbers, and custom fields) and stamps it directly into the object's passport slot. For Seurat objects the passport is stored in `@misc$passport`; for `SingleCellExperiment` and `SummarizedExperiment` objects it is stored in `metadata(obj)$passport`. All information persists inside the object when saved as `.rds`.

If a passport already exists in the object, all previously filled fields are pre-loaded into the popup so you only need to update what changed.

Usage

```
scPassport(obj, parent = NULL, read = FALSE)
```

Arguments

`obj`	A Seurat, SingleCellExperiment, or SummarizedExperiment object to stamp. Must not be NULL.
`parent`	A parent object of the same class that this object was subset from. If provided, `parent_id` and `lineage` are automatically populated from the parent's passport - no typing needed. Default is `NULL` (root object).
`read`	Logical. If `TRUE`, prints the existing passport to the console instead of opening the popup. Default is `FALSE`.

Details

The passport contains the following fields:

Identity:

- `object_id` -Name/ID of this object (e.g. "WTHeme")
- `rds_self` -RDS registry number of this object (e.g. 224)
- `created` -Timestamp of when passport was stamped

Animal Info:

- `animal_id` -Individual animal identifier (e.g. "M01")
- `species` -Species name (e.g. "Mus musculus")

- sex -Sex of the animal (e.g. "male", "female")
- age -Age of the animal (e.g. "P60")
- condition -Experimental condition (e.g. "control", "treated")
- tissue -Tissue of origin (e.g. "prefrontal cortex")

Experiment Info:

- project -Project name (e.g. "memory_study")
- researcher -Name of the researcher
- date -Date of the experiment
- notes -Any free-text notes

Lineage:

- parent_id -Object ID of the direct parent (or "root")
- rds_parent -RDS number of the parent object (or "root")
- lineage -Full ancestry chain as a character vector
- children -Object IDs of children subset from this object
- rds_children -RDS numbers of children objects

Custom Fields: Any additional fields typed into the Custom Fields section of the popup are automatically appended to the passport and displayed when read = TRUE.

Safety: The function will never overwrite your object with NULL. Cancelling the popup or any error returns the original object unchanged.

Value

The same object with its passport slot filled in. If cancelled or an error occurs, the original object is returned unchanged

- it will never return NULL.

Main Functions

- scPassport — Open popup to fill/update passport
- read_passport — Print passport to console
- log_step — Log a processing step

Typical Workflow

```
# 1. Stamp your root object
WTHeme <- scPassport(WTHeme)

# 2. Log processing steps
WTHeme <- NormalizeData(WTHeme)
WTHeme <- log_step(WTHeme, "NormalizeData")
```

```
# 3. Subset and stamp child, linking to parent
EndofrHeme <- subset(WTHeme, subset = cell_type == "Endothelial")
EndofrHeme <- scPassport(EndofrHeme, parent = WTHeme)

# 4. Read passport anytime
read_passport(EndofrHeme)
```

Author(s)

Maintainer: Sedat Kacar <sedatkacar56@gmail.com> (ORCID)

See Also

Useful links:

- https://github.com/sedatkacar56/scPassport
- Report bugs at https://github.com/sedatkacar56/scPassport/issues

read_passport, log_step

Examples

```
# Read passport on an unstamped object (prints "No passport found")
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    se <- SummarizedExperiment::SummarizedExperiment()
    scPassport(se, read = TRUE)
}


# Stamp a root object (requires an existing Seurat object 'WTHeme')
# WTHeme <- scPassport(WTHeme)

# Stamp a child subset, linking to parent automatically
# EndofrHeme <- scPassport(EndofrHeme, parent = WTHeme)

# Read existing passport without opening popup
# scPassport(WTHeme, read = TRUE)

# Or use the dedicated read function
# read_passport(WTHeme)
```

---

validate_passport *Check whether a passport list is valid*

Description

Check whether a passport list is valid

Usage

```
validate_passport(passport)
```

Arguments

passport	A list (the `@misc$passport` slot of a Seurat object).

Value

TRUE if the passport contains all required fields, FALSE otherwise.

Index

∗ **internal**
append_log_entry, 2
build_log_entry, 3
build_passport, 3
validate_passport, 9

append_log_entry, 2

build_log_entry, 3
build_passport, 3

log_step, 5, *6*, *8, 9*

read_passport, *5*, 6, *8, 9*

scPassport, *5, 6*, 7, *8*
scPassport-package *(*scPassport*)*, 7

validate_passport, 9