

Package ‘alabaster.se’

May 30, 2023

Title Load and Save SummarizedExperiments from File

Version 1.0.0

Date 2023-02-20

License MIT + file LICENSE

Description Save SummarizedExperiments into file artifacts, and load them back into memory.
This is a more portable alternative to serialization of such objects into RDS files.
Each artifact is associated with metadata for further interpretation;
downstream applications can enrich this metadata with context-specific properties.

Depends SummarizedExperiment, alabaster.base

Imports methods, alabaster.ranges, alabaster.matrix, BiocGenerics,
S4Vectors, IRanges, GenomicRanges

Suggests rmarkdown, knitr, testthat, BiocStyle, jsonlite

VignetteBuilder knitr

RoxygenNote 7.2.3

biocViews DataImport, DataRepresentation

git_url <https://git.bioconductor.org/packages/alabaster.se>

git_branch RELEASE_3_17

git_last_commit e7fc40c

git_last_commit_date 2023-04-25

Date/Publication 2023-05-30

Author Aaron Lun [aut, cre]

Maintainer Aaron Lun <infinite.monkeys.with.keyboards@gmail.com>

R topics documented:

emptyRowRanges	2
loadSummarizedExperiment	2
stageObject,SummarizedExperiment-method	3

Index	6
--------------	----------

emptyRowRanges	<i>Is the rowRanges empty?</i>
----------------	--------------------------------

Description

Check the `rowRanges` of a `RangedSummarizedExperiment` is empty, i.e., a `GRangesList` with no ranges.

Usage

```
emptyRowRanges(x)
```

Arguments

`x` A `RangedSummarizedExperiment` object or the contents of its `rowRanges`.

Details

Metadata in `mcols` is ignored for the purpose of this discussion, as this can be moved to the `rowData(x)` of the base `SummarizedExperiment` class without loss. In other words, non-empty `mcols` will not be used to determine that the `rowRanges` is not empty. However, non-empty fields in the `metadata` or in the inner `mcols` of the `GRanges` will trigger a non-emptiness decision.

Value

A logical scalar indicating whether `x` has empty `rowRanges`.

Examples

```
emptyRowRanges(SummarizedExperiment())
emptyRowRanges(SummarizedExperiment(rowRanges=GRanges()))
emptyRowRanges(SummarizedExperiment(rowRanges=GRangesList()))
```

loadSummarizedExperiment	<i>Load a SummarizedExperiment</i>
--------------------------	------------------------------------

Description

Default loading of `SummarizedExperiments` based on the metadata stored by the corresponding `stageObject` method.

Usage

```
loadSummarizedExperiment(exp.info, project)
```

Arguments

exp.info Named list containing the metadata for this experiment.

project Any argument accepted by the acquisition functions, see [?acquireFile](#). By default, this should be a string containing the path to a staging directory.

Value

A [SummarizedExperiment](#) or [RangedSummarizedExperiment](#) object.

Author(s)

Aaron Lun

Examples

```
# Mocking up an experiment:
mat <- matrix(rpois(10000, 10), ncol=10)
colnames(mat) <- letters[1:10]
rownames(mat) <- sprintf("GENE_%i", seq_len(nrow(mat)))

se <- SummarizedExperiment(list(counts=mat))
se$stuff <- LETTERS[1:10]
rowData(se)$blah <- runif(1000)
metadata(se)$whee <- "YAY"

# Staging it:
tmp <- tempfile()
dir.create(tmp)
info <- stageObject(se, dir=tmp, "rna-seq")

# And loading it back in:
loadSummarizedExperiment(info, tmp)
```

stageObject, SummarizedExperiment-method
Stage a SummarizedExperiment

Description

Save a [SummarizedExperiment](#) to file inside the staging directory.

Usage

```
## S4 method for signature 'SummarizedExperiment'
stageObject(x, dir, path, child = FALSE, meta.name = "experiment.json", ...)

## S4 method for signature 'RangedSummarizedExperiment'
stageObject(x, dir, path, child = FALSE, ..., skip.ranges = FALSE)
```

Arguments

<code>x</code>	A SummarizedExperiment object or one of its subclasses.
<code>dir</code>	String containing the path to the staging directory.
<code>path</code>	String containing a prefix of the relative path inside <code>dir</code> where <code>x</code> is to be saved. The actual path used to save <code>x</code> may include additional components, see Details .
<code>child</code>	Logical scalar indicating whether <code>x</code> is a child of a larger object.
<code>meta.name</code>	String containing the name of the metadata file.
<code>...</code>	Further arguments to pass to the SummarizedExperiment method. For the <code>SummarizedExperiment</code> itself, all further arguments are just ignored.
<code>skip.ranges</code>	Logical scalar indicating whether to avoid saving the rowRanges .

Details

`meta.name` is only needed to set up the output path, for consistency with the [stageObject](#) contract. Callers should make sure to write the metadata to the same path by using [.writeMetadata](#) to create the JSON file.

If `skip.ranges=TRUE`, the `RangedSummarizedExperiment` method just calls the `SummarizedExperiment` method, i.e., [rowRanges](#) are not saved. This avoids the hassle of switching classes and the associated problems, e.g., <https://github.com/Bioconductor/SummarizedExperiment/issues/29>. Note that any subsequent [loadObject](#) call on the staged assets will return a non-ranged `SummarizedExperiment`.

If `x` is a `RangedSummarizedExperiment` with “empty” [rowRanges](#) (i.e., a [GRangesList](#) with zero-length entries), `stageObject` will save it to file without any genomic range information. This means that any subsequent [loadObject](#) on the staged assets will return a non-ranged `SummarizedExperiment`.

By default, we consider the presence of data frames in the assays to be an error. Users should coerce these into an appropriate matrix type, e.g., a dense matrix or a sparse `dgCMatrx`. If a `DataFrame` as an assay is truly desired, users may set `options(alabaster.se.reject_data.frames=FALSE)` to skip the error. Note that this only works for [DataFrame](#) objects - `data.frame` objects will not be saved correctly.

Value

A named list of metadata that follows the `summarized_experiment` schema. The contents of `x` are saved into a path subdirectory inside `dir`.

Author(s)

Aaron Lun

Examples

```
tmp <- tempfile()
dir.create(tmp)

mat <- matrix(rpois(10000, 10), ncol=10)
```

```
colnames(mat) <- letters[1:10]
rownames(mat) <- sprintf("GENE_%i", seq_len(nrow(mat)))

se <- SummarizedExperiment(list(counts=mat))
se$stuff <- LETTERS[1:10]
rowData(se)$blah <- runif(1000)
metadata(se)$whee <- "YAY"

dir.create(tmp)
stageObject(se, dir=tmp, "rna-seq")
list.files(file.path(tmp, "rna-seq"))
```

Index

.writeMetadata, [4](#)

acquireFile, [3](#)

DataFrame, [4](#)

emptyRowRanges, [2](#)

GRanges, [2](#)

GRangesList, [2](#), [4](#)

loadObject, [4](#)

loadSummarizedExperiment, [2](#)

mcols, [2](#)

metadata, [2](#)

options, [4](#)

RangedSummarizedExperiment, [2](#), [3](#)

rowData, [2](#)

rowRanges, [2](#), [4](#)

stageObject, [2](#), [4](#)

stageObject, RangedSummarizedExperiment-method
(stageObject, SummarizedExperiment-method),
[3](#)

stageObject, SummarizedExperiment-method,
[3](#)

SummarizedExperiment, [2–4](#)