

Package: recordr (via r-universe)

October 1, 2026

Title R Provenance Tracking

Version 1.0.3.9000

Description Provide methods to record data provenance about R script executions. Provenance data includes files that were read and written by the script, along with information about the execution, such as start time end time, the R modules loaded during the execution, and other information describing the execution environment.

Date 2016-08-31

Depends R (>= 3.1.1), methods

Imports uuid, datapack, digest, XML, RSQLite, rappdirs, hash, EML

Suggests dataone, ggplot2, knitr, png, raster, readr, rgdal, rmarkdown, testthat, DiagrammeR

VignetteBuilder knitr

License Apache License 2.0

URL <https://github.com/NCEAS/recordr>

BugReports <https://github.com/NCEAS/recordr/issues>

Collate 'Constants.R' 'Recordr.R' 'ExecMetadata.R' 'FileMetadata.R' 'ProvRels.R' 'onLoad.R' 'recordr-package.R' 'tracedFunctions.R'

RoxygenNote 6.0.1

Config/pak/sysreqs cmake make libjq-dev libuv1-dev libxml2-dev libssl-dev librdf0-dev libnode-dev

Repository <https://dataoneorg.r-universe.dev>

Date/Publication 2018-10-30 22:46:36 UTC

RemoteUrl <https://github.com/NCEAS/recordr>

RemoteRef HEAD

RemoteSha f926eb0f75480fb30a93a26b7071c91768750b7d

Contents

changeHome	3
coverageElement	3
deleteRuns	4
endRecord	5
ExecMetadata-class	5
FileMetadata-class	7
geoCoverage	8
getDBconnection	8
getMetadata	9
initialize,ExecMetadata-method	9
initialize,FileMetadata-method	11
initialize,ProvRels-method	12
initialize,Recordr-method	13
listRuns	13
makeEML	15
plotRuns	16
ProvRels-class	17
publishRun	18
putMetadata	19
readExecMeta	20
readFileMeta	21
readProvRels	22
record	23
recordr	24
Recordr-class	24
recordr_createObject	25
recordr_getObject	26
recordr_ggsave	26
recordr_raster	26
recordr_read.csv	27
recordr_read_csv	27
recordr_readLines	28
recordr_readOGR	28
recordr_readPNG	28
recordr_scan	29
recordr_updateObject	29
recordr_write.csv	29
recordr_write_csv	30
recordr_writeLines	30
recordr_writeOGR	30
recordr_writePNG	31
recordr_writeRaster	31
selectRuns	32
standardizeCall	33
startRecord	33
traceRuns	34

<i>changeHome</i>	3
unArchiveFile	36
updateExecMeta	36
upgradeRecordr	37
viewRuns	38
writeExecMeta	39
writeFileMeta	40
writeProvRel	40
Index	42

changeHome	<i>Change the recordr home directory</i>
------------	--

Description

Change the recordr home directory

Usage

changeHome(recordr, currentDir, newDir = as.character(NA), copy, ...)

Arguments

- recordr A recordr object
- currentDir A character value specifying the current recordr home directory
- newDir A character value, specifying the new recordr home directory
- copy A logical value. A value of TRUE causes data to be copied from the old
- ... Additional arguments directory to the new one. A default value is not set.

coverageElement	<i>Create a coverage element</i>
-----------------	----------------------------------

Description

Create a coverage element

Usage

coverageElement(gc, tempc)

Arguments

- gc An EML::geographicCoverage object
- tempc A EML::temporalCoverage object

Value

An EML::Coverage object

deleteRuns	<i>Delete runs that match search parameters</i>
------------	---

Description

The execution metadata and all archived files associated with each matching run are permanently deleted from the file system. No backup is maintained by the recordr package, so this deletion is irreversible, unless the user maintains their own backup.

Usage

```
deleteRuns(recordr, ...)
```

```
## S4 method for signature 'Recordr'
deleteRuns(recordr, id = as.character(NA),
  file = as.character(NA), start = as.character(NA),
  end = as.character(NA), tag = as.character(NA),
  error = as.character(NA), seq = as.integer(NA), noop = FALSE,
  quiet = FALSE)
```

Arguments

recordr	A Recordr instance
...	additional arguments
id	An execution identifier
file	The name of script to match.
start	A one or two element character list specifying a date range to match for run start time
end	A one or tow element character list specifying a date range to match for run end time
tag	The text of the tags to match.
error	The text of the error message to match.
seq	The run sequence number (can be a single value or a range, e.g seq="1:10")
noop	Don't delete any date, just show what would be deleted.
quiet	A logical if TRUE then output is not printed. Useful if only the return value is desired.

Value

A data.frame containing execution metadata for the runs that were deleted.

See Also

[Recordr](#) class description

endRecord	<i>End the recording session that was started by startRecord()</i>
-----------	--

Description

The recording session started by the startRecord() method is terminated and all provenance collecting is discontinued. A log of all the console commands is saved.

Usage

```
endRecord(recordr)

## S4 method for signature 'Recordr'
endRecord(recordr)
```

Arguments

recordr A Recordr instance

Value

id The execution identifier that uniquely identifies this execution.

See Also

[Recordr](#) class description

Examples

```
## Not run:
rc <- new("Recordr")
startRecord(rc, tag="my first console run")
x <- read.csv(file="./test.csv")
runIdentifier <- endRecord(rc)

## End(Not run)
```

ExecMetadata-class	<i>A class representing a script execution with the run manager</i>
--------------------	---

Description

A class representing a script execution with the run manager

Slots

executionId A character containing the unique identifier for this execution.

metadataId A character containing the unique identifier for the associated metadata object.

tag A character vector containing text associated with this execution.

datapackageId A character containing the unique identifier for an uploaded package.

user A character containing the user name that ran the execution.

subject A character containing the user identity that uploaded the package.

hostId A character containing the host identifier to which the package was uploaded.

startTime A character containing the start time of the execution.

operatingSystem A character containing the operating system name.

runtime A character containing R build and version information.

softwareApplication A character containing the software application used, e.g. ("R")

moduleDependencies A character containing the modules used by the software application.

endTime A character containing the end time of the execution.

errorMessage A character containing any error messages captured during the execution.

publishTime A character containing the time that the execution package was uploaded to a repository.

publishNodeId A character containing the node name that the execution was published to.

publishId A character containing the identifier for the uploaded package.

console A logical indicating whether this was a console session, i.e. startRecord() -> endRecord()

seq A integer containing a simple integer value associated with the execution.

Methods

- [initialize](#): Initialize an execution metadata object
- [readExecMeta](#): Retrieve saved Execution metadata.
- [writeExecMeta](#): Save a single execution metadata.
- [updateExecMeta](#): Update saved execution metadata.

Author(s)

slaughter

See Also

[recordr](#) package description.

FileMetadata-class	<i>A class containing information about a file or group of files</i>
--------------------	--

Description

A class containing information about a file or group of files

Details

This class is used internally by the recordr package.

Slots

fileId a character containing the unique identifier for the file entry
executionId a character containing the identifier associated with the file entry
filePath a character containing the location of the file
sha256 a character containing the check of the file
size a numeric containing the size of the file
user a character containing the user associated with the file entry.
createTime a character containing the file creation time.
modifyTime a character containing the file modification time.
access a character containing the type of access made to the file ("read", "write", "execute")
format a character containing the file format (e.g. "text/csv")
archivedFilePath a character containing the location of the archived file

Methods

- [initialize](#): Initialize a FileMetadata object
- [readFileMeta](#): Retrieve saved file metadata for one or more files
- [writeFileMeta](#): Save metadata for a single file.

See Also

[recordr](#) package description.

geoCoverage	<i>Create a geographic coverage element from a description and bounding coordinates</i>
-------------	---

Description

Create a geographic coverage element from a description and bounding coordinates

Usage

```
geoCoverage(geoDescription, west, east, north, south)
```

Arguments

geoDescription	a character string containing the description of the geogragraphic coverage
west	a character string containing the western most coordinate of the coverage (ex. "-134.32")
east	a character string containing the eastern most coordinate of the coverage (ex. "-120.42")
north	a character string containing the northern most coordinate of the coverage (ex. "34.32")
south	a character string containing the southern ost coordinate of the coverage (ex. "30.14")

getDBconnection	<i>Get a database connection</i>
-----------------	----------------------------------

Description

Get a database connection

Usage

```
getDBconnection(dbFile)
```

Arguments

dbFile	the path to the recordr database file (default: ~/.recordr/recordr.sqlite)
--------	--

getMetadata

*Retrieve the metadata object for a run***Description**

When a script or console session is recorded (see `record()` and `startrecord()`), a metadata object is created that describes the objects associated with the run, using the Ecological Metadata Language <https://knb.ecoinformatics.org/#external//emlparser/docs/index.html>. This metadata can be retrieved from the `recordr` cache for review or editing if desired. If the metadata is updated, it can be re-inserted into the `recordr` cache using the `putMetadata` method.

Usage

```
getMetadata(recordr, ...)

## S4 method for signature 'Recordr'
getMetadata(recordr, id = as.character(NA),
  seq = as.character(NA), as = as.character("text"))
```

Arguments

<code>recordr</code>	a <code>Recordr</code> instance
<code>...</code>	additional parameters see also Recordr class description
<code>id</code>	The identifier for a run
<code>seq</code>	The sequence number for a run
<code>as</code>	Form to return the metadata as. Possible values are: "text", "parsed" (for parsed XML), or "EML" (for an EML R package S4 object)

Value

A character vector containing the metadata

initialize, ExecMetadata-method

*Initialize an execution metadata object***Description**

Initialize an execution metadata object

Usage

```
## S4 method for signature 'ExecMetadata'
initialize(.Object, executionId = as.character(NA),
  metadataId = as.character(NA), tag = as.character(NA),
  datapackageId = as.character(NA), user = as.character(NA),
  subject = as.character(NA), hostId = as.character(NA),
  startTime = as.character(NA), operatingSystem = as.character(NA),
  runtime = as.character(NA), moduleDependencies = as.character(NA),
  programName = as.character(NA), endTime = as.character(NA),
  errorMessage = as.character(NA), publishTime = as.character(NA),
  publishNodeId = as.character(NA), publishId = as.character(NA),
  console = FALSE, seq = as.integer(0))
```

Arguments

.Object	The ExecMetada object
executionId	a "character", the unique identifier for an execution
metadataId	a "character", the unique identifier for the metadata object associated with an execution
tag	A character vector that describes this execution.
datapackageId	a "character", the unique identifier for the datapackage associated with an execution
user	a "character", the user that started the execution
subject	a "character", the user identity that owns the uploaded execution datapackage
hostId	a "character", the host identifier that the execution datapackage was uploaded to
startTime	a "character", the starting time of the execution
operatingSystem	a "character", the operating system that the execution ran on
runtime	a "character", the software application used for the run, e.g. "R version 3.2.3 (2015-12-10)"
moduleDependencies	a "character" vector, a list of modules loaded during an execution
programName	a "character", The name of the program that is being run.
endTime	a "character", the ending time of an execution
errorMessage	a "character", error messages generated during an execution
publishTime	a "character", the time of publication (uploading) of an execution package
publishNodeId	a "character", the node identifier that an execution package was published to
publishId	a "character", the unique identifier associated with a published execution
console	a "logical", was this execution recorded as commands typed at the console
seq	an "integer", an integer associated with an execution

See Also

[ExecMetadata](#) class description

initialize,FileMetadata-method

Initialize a file metadata object.

Description

Initialize a file metadata object.

Usage

```
## S4 method for signature 'FileMetadata'
initialize(.Object, file, fileId = as.character(NA),
  sha256 = as.character(NA), size = as.numeric(0),
  user = as.character(NA), createTime = as.character(NA),
  modifyTime = as.character(NA), executionId, access = as.character(NA),
  format = as.character(NA), archivedFilePath = as.character(NA))
```

Arguments

.Object	a "FileMetdata" object
file	a "character", a file to acquire metadata for
fileId	a "character", the unique identifier for this FileMeta object
sha256	a "character", the checksum for the file
size	a "numeric", size in bytes of the file
user	a "character", the user that owns the file
createTime	a "character", the creation time of the file
modifyTime	a "character", the modification time of the file
executionId	a "character", the executionId associated with this FileMeatadata object
access	"character", the access that occurred for this file ("read", "write", "execute")
format	a "character", the format type associate with the file, e.g. "text/csv"
archivedFilePath	a "character", the file path of the file

Details

This method is used internally by the recordr package.

See Also

[FileMetadata](#) class description

`initialize,ProvRels-method`*Initialize a provenance relationship object.*

Description

Initialize a provenance relationship object.

Usage

```
## S4 method for signature 'ProvRels'
initialize(.Object, executionId = as.character(NA),
  subject = as.character(NA), predicate = as.character(NA),
  object = as.character(0), subjectType = as.character(NA),
  objectType = as.character(NA), dataTypeURI = as.character(NA))
```

Arguments

<code>.Object</code>	a "ProvRels" object
<code>executionId</code>	a "character", the execution identifier for an execution
<code>subject</code>	a "character", the<BS>subject of the provenance relationship
<code>predicate</code>	a "character", the predicate of the provenance relationship
<code>object</code>	a "character", the object of the provenance relationship
<code>subjectType</code>	a "character", the RDF node type of the subject, i.e. "url", "blank"
<code>objectType</code>	a "character", the RDF node type of the object i.e. "url", "blank"
<code>dataTypeURI</code>	a "character", the RDF data type of the object, i.e. "xsd:string"

Details

This method is used internally by the recordr package.

See Also

[ProvRels](#) class description

initialize, Recordr-method

Initialize a Recorder object

Description

Initialize a Recorder object

Usage

```
## S4 method for signature 'Recordr'
initialize(.Object, newDir = as.character(NA),
  copy = TRUE, ...)
```

Arguments

.Object	The Recordr object
newDir	The recordr home directory is changed to the new location.
copy	A logical value: if TRUE and newDir is specified, then copy data from the existing recordr home to the new one.
...	Additional parameters

Details

A recordr object is returned that can be used with other recordr package methods. When the optional newDir argument is used, the recordr home directory is changed to the new value. The default behaviour is to have data copied from the old home directory to the new one, but this can be changed by using the copy argument, i.e. See the recordr vignette 'recordr Package Introduction' for more information about information that recordr stores in the recordr home directory.

See Also

[Recordr](#) class description

listRuns

List all runs recorded by record() or startRecord()

Description

If no search terms are specified, then all runs are listed. The method arguments are search terms that limit the runs listed, with any runs listed that match all arguments.

Usage

```
listRuns(recordr, ...)

## S4 method for signature 'Recordr'
listRuns(recordr, id = as.character(NA),
  script = as.character(NA), start = as.character(NA),
  end = as.character(NA), tag = as.character(NA),
  error = as.character(NA), seq = as.character(NA),
  orderBy = "-startTime", quiet = FALSE, full = FALSE)
```

Arguments

<code>recordr</code>	A <code>Recordr</code> instance
<code>...</code>	additional parameters
<code>id</code>	a "character", the identifier to match
<code>script</code>	"character", the name of the script to match
<code>start</code>	"character", Match runs that started in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
<code>end</code>	a "character", Match runs that ended in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
<code>tag</code>	"character" Text of tag to match
<code>error</code>	"character" Text of error message to match
<code>seq</code>	"integer" A run sequence number (can be a range, e.g. <code>seq=1:10</code>)
<code>orderBy</code>	The column that will be used to sort the output. This can include a minus sign before the name, e.g. <code>-startTime</code>
<code>quiet</code>	A logical, if TRUE then output is not printed to the console. Default is FALSE.
<code>full</code>	A logical, if TRUE then all output columns are printed, regardless of console width.

Details

The "start" and "end" parameters can be used to specify a time range to find runs that started execution and ended in the specified time range. For examples, specifying `"start=c("2015-01-01", "2015-01-31")` will cause the search to return any execution with a starting time in the first month of 2015.

Value

data frame containing information for each run

See Also

[Recordr](#) class description

Examples

```
## Not run:
rc <- new("Recordr")
# List runs that started in January 2015
listRuns(rc, start=c("2015-01-01", "2015-01-31"))
# List runs that started on or after March 1, 2014
listRuns(rc, start="2014-03-01")
# List runs that contain a tag with the string "analysis v1.3"
listRuns(rc, tag="analysis v1.3")

## End(Not run)
```

makeEML	Create a minimal EML document.
---------	--------------------------------

Description

An EML document is create from the values passed in.

Usage

```
makeEML(recordr, id, system, title, creators, abstract = NA,
  methodDescription = NA, geo_coverage = NA, temp_coverage = NA,
  endpoint = NA)
```

Arguments

recordr	A Recordr object.
id	The identifier for the EML document.
system	The system for the document.
title	The document title.
creators	A list of creator elements.
abstract	The document abstract.
methodDescription	The dataset method description.
geo_coverage	The geographic coverage element.
temp_coverage	The temporal coverage element.
endpoint	The online distribution URL.

plotRuns

Trace processing lineage for a run and plot it.

Description

A data processing workflow might include multiple processing steps, with each step being performed by a separate R script. These multiple steps are linked by the files that one step writes and the next step in the workflow reads. The plotRuns method finds these connections between executions to determine the executions that comprise a processing workflow.

Usage

```
plotRuns(recordr, ...)

## S4 method for signature 'Recordr'
plotRuns(recordr, id = as.character(NA),
  file = as.character(NA), start = as.character(NA),
  end = as.character(NA), tag = as.character(NA),
  error = as.character(NA), seq = as.character(NA),
  orderBy = "-startTime", direction = "both", quiet = TRUE, ...)
```

Arguments

recordr	a Recordr instance
...	additional parameters
id	The identifier for a run. Either id or seq can be specified, not both.
file	The name of script to match
start	Match runs that started in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
end	Match runs that ended in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
tag	The text of tag to match
error	The text of error message to match.
seq	The sequence number for a run. #' @param id The execution identifier of a run to view
orderBy	Sort the results according to the specified column. A hyphen ('-') prepended to the column name denotes a descending sort. The default value is "-startTime"
direction	The direction to trace the lineage, either forward, backward, or both. The default is both
quiet	A logical if TRUE then output is not printed.

Details

If the run id or seq number is know for the run to be traced, then one or the other of these values can be used. Alternatively, other run attributes can be used to determine the run to be traced, such as file, start, etc. If these other search parameters are used and multiple runs are selected, only the first run selected will be traced. These search parameters can be used together to easily find certain runs, for example, the latest run of a particular script, the latest run with a specified tag specified, etc. (see examples).

Value

A list of the execution identifiers that are in the processing workflow.

See Also

[Recordr](#) class description

Examples

```
## Not run:
# Plot processing workflow for the run with sequence number '101'
plotRuns(recordr, seq=101)
# Plot processing workflow for the last execution of script "runModel.R"
plotRuns(recordr, file="runModel.R", orderBy="-startTime")
# Plot processing workflow for the last execution with the tag 'best run yet!' specified.
plotRuns(recordr, tag="best run yet!", orderBy="-startTime")

## End(Not run)
```

ProvRels-class

A class containing information about a file or group of files

Description

A class containing information about a file or group of files

Details

This class is used internally by the recordr package.

Slots

executionId a character containing the identifier associated with the file entry
 subject a character containing the subject of a provenance relationship
 predicate a character containign the predicate of a provenance relationship
 object a character containing the object of a provenance relationship
 subjectType, a character containing the RDF node type of the the subject, values can be 'uri',
 'blank'

objectType a character containign the RDF node type of the object, each value can be 'uri', 'blank', or 'literal'

dataTypeURI The RDF data type that specifies the type of the object

Methods

- [initialize](#): Initialize a ProvRels object
- [readProvRels](#): Retrieve saved provenance relationships.
- [writeProvRel](#): Save a provenance relationship.object

See Also

[recordr](#) package description.

publishRun

Publish a recordr'd execution to DataONE

Description

Publish a recordr'd execution to DataONE

Usage

```
publishRun(recordr, ...)
```

```
## S4 method for signature 'Recordr'
publishRun(recordr, id = as.character(NA),
  seq = as.character(NA), assignDOI = FALSE, update = FALSE,
  quiet = TRUE, retPkg = FALSE)
```

Arguments

recordr	a Recordr instance
...	additional parameters seealso Recordr class description
id	the run identifier for the execution to upload to DataONE
seq	The sequence number for the execution to upload to DataONE
assignDOI	a boolean value: if TRUE, assign DOI values for system metadata, otherwise assign uuid values
update	a boolean value: if TRUE, republish a previously published execution
quiet	A boolean value: if TRUE, informational messages are not printed (default=TRUE)
retPkg	A boolean value: if TRUE, then the package that was uploaded is returned, if FALSE then the identifier of the package is returned (default=FALSE).

Value

The published identifier of the uploaded package

putMetadata*Update the metadata object for a run*

Description

Put a metadata document into the recordr cache for an run, replacing the existing metadata object for the specified run, if one exists.

Usage

```
putMetadata(recordr, ...)  
  
## S4 method for signature 'Recordr'  
putMetadata(recordr, id = as.character(NA),  
  seq = as.character(NA), metadata = as.character(NA), asText = TRUE)
```

Arguments

recordr	a Recordr instance
...	additional parameters
id	The identifier for a run
seq	The sequence number for a run
metadata	The replacement metadata, as the actual text, or as a filename containing the metadata
asText	A logical. See 'Details'. If TRUE, then the metadata parameter is a character vector containing metadata, if FALSE it is a filename. The default is TRUE.

Details

The metadata parameter can specify either a character vector that contains the metadata this parameter can be a filename that contains the metadata. The asText parameter is used to specify which type of value is specified. If asText is TRUE, then the metadata parameter is a character vector, if it is FALSE, then the metadata parameter is a filename.

Value

A character vector containing the metadata

See Also

[Recordr](#) class description

readExecMeta	<i>Retrieve saved execution metadata.</i>
--------------	---

Description

Execution metadata is retrived from recordr database table `_execmeta_` based on search parameters.

Usage

```
readExecMeta(recordr, ...)

## S4 method for signature 'Recordr'
readExecMeta(recordr, executionId = as.character(NA),
  script = as.character(NA), startTime = as.character(NA),
  endTime = as.character(NA), tag = as.character(NA),
  errorMessage = as.character(NA), seq = as.integer(NA),
  orderBy = as.character(NA), sortOrder = "ascending", delete = FALSE,
  ...)
```

Arguments

<code>recordr</code>	A Recordr object
<code>...</code>	additional parameters
<code>executionId</code>	A character value that specifies an execution identifier to search for.
<code>script</code>	A character value that specifies a script name to search for.
<code>startTime</code>	A character value that specifies the start of a time range. This value must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to "YYYY-MM-DD"
<code>endTime</code>	A character value that specifies the end of a time to to search. This value must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to "YYYY-MM-DD"
<code>tag</code>	A tag value to search for
<code>errorMessage</code>	An execution error message to search for.
<code>seq</code>	An exectioin sequence nuber
<code>orderBy</code>	The column to sort the result set by.
<code>sortOrder</code>	The sort order. Values include "ascending", "descending".
<code>delete</code>	a "logical", if TRUE, the selected runs are deleted (default: FALSE).

Details

The `"startTime"` and `"endTime"` parameters are used to specify a time range to find runs that started execution between the start and end times that are specified.

Value

A list of ExecMetadata objects

See Also

[ExecMetadata](#) class description

readFileMeta	<i>Retrieve saved file metadata for one or more files</i>
--------------	---

Description

File metadata is retrived from the recordr database table *filemeta* based on search parameters.

Usage

```
readFileMeta(recordr, ...)

## S4 method for signature 'Recordr'
readFileMeta(recordr, fileId = as.character(NA),
  executionId = as.character(NA), filePath = as.character(NA),
  sha256 = as.character(NA), user = as.character(NA),
  access = as.character(NA), format = as.character(NA),
  orderBy = as.character(NA), sortOrder = "ascending", delete = FALSE,
  ...)
```

Arguments

recordr	A recordr object
...	Additional parameters
fileId	The id of the file to search for
executionId	A character value that specifies an execution identifier to search for.
filePath	The path name of the file to search for.
sha256	The sha256 checksum value for the uncompressed file.
user	The user that ran the execution that created or accessed the file.
access	The type of access for the file. Values include "read", "write", "execute"
format	The format type of the object, e.g. "text/plain"
orderBy	The column to sort the result set by.
sortOrder	The sort type. Values include ("ascending", "descending")
delete	a "logical", if TRUE, the selected file entries are deleted (default: FALSE).

Details

This method is used internally by the recordr package.

Value

A dataframe containing file metadata objects

See Also

[FileMetadata](#) class description

readProvRels

Retrieve saved file metadata for one or more files

Description

File metadata is retrived from the recordr database table *filemeta* based on search parameters.

Usage

```
readProvRels(recordr, ...)

## S4 method for signature 'Recordr'
readProvRels(recordr, executionId = as.character(NA),
  subject = as.character(NA), predicate = as.character(NA),
  object = as.character(NA), subjectType = as.character(NA),
  objectType = as.character(NA), dataTypeURI = as.character(NA),
  orderBy = as.character(NA), sortOrder = "ascending", delete = FALSE,
  ...)
```

Arguments

recordr	A recordr object
...	Additional parameters
executionId	A character value that specifies an execution identifier to search for.
subject	The subject of the provenance relationships to match
predicate	The predicate of the provenance relationships to match
object	The object of the provenance relationships to match
subjectType	A character value containing the subject type of the relationship to match
objectType	A character value containing the object type of the relationship to match
dataTypeURI	A character value containing the data type of the relationship to match
orderBy	The column to sort the result set by.
sortOrder	The sort type. Values include ("ascending", "descending")
delete	a "logical", if TRUE, the selected file entries are deleted (default: FALSE).

Details

This method is used internally by the recordr package.

Value

A dataframe containing file metadata objects

See Also

[ProvRels](#) class description

record	<i>Record data provenance for an R script execution</i>
--------	---

Description

The R script is executed and information about file reads and writes is recorded.

Usage

```
record(recordr, file, ...)

## S4 method for signature 'Recordr'
record(recordr, file, tag = "", ...)
```

Arguments

recordr	a Recordr instance
file	The name of the R script to run and collect provenance information for
...	additional parameters that will be passed to the R "base::source()" function
tag	A string that will be associated with this run

Details

Input files, the script itself and igenerated files are archived. Information about the execution environment is also saved.

Value

The execution identifier for this run

See Also

[Recordr](#) class description

Examples

```
## Not run:
rc <- new("Recordr")
executionId <- record(rc, file="myscript.R", tag="first run of myscript.R")

## End(Not run)
```

recordr	<i>Record, review and publish data provenance.</i>
---------	--

Description

The R package *recordr* provides methods to easily record data provenance about R script executions, such as the files that were read and written by the script, along with information about the execution, such as start time end time, the R modules loaded during the execution, etc. This provenance information along with any files created by the script can then be combined into a data package and uploaded to a data repository such as DataONE.

Details

An overview of the recordr package is available with the R command: `'vignette("recordr_overview")'`.

Classes

- [Recordr](#): A class containing methods to record, review and publish data provenance

Author(s)

Peter Slaughter (NCEAS), Matthew B. Jones (NCEAS), Christopher Jones (NCEAS)

Examples

```
## Not run:
# This example shows how to record provenance for an R script and view the recorded
# information.
library(recordr)
rc <- new("Recordr")
record(rc, "./myScript.R", tag="Simple script recording #1")
listRuns(rc, tag="recording #1")
viewRuns(rc, tag="recording #1")

## End(Not run)
```

Recordr-class	<i>Capture, review and publish data provenance</i>
---------------	--

Description

The *Recordr* class provides methods to record, search, review and publish data provenance about R script executions. Information about files read and written by a script and the execution environment can be captured for each script execution. Script executions can then be reviewed and selected to be published to the DataONE data repository, by retrieving archived copies of the R script, the files read and written by a script and a description of the provenance relationships between objects in the run, which are then combined into a package and uploaded to the requested member node.

Slots

recordrDir value of type "character" containing a path to the Recordr working directory

dbConn A value of type "SQLiteConnection" that contains the connection of the recordr database

dbFile A value of type "character" that contains the location of the recordr database file

Methods

- [initialize](#): Initialize a Recordr object
- [startRecord](#): Begin recording provenance for an R session
- [endRecord](#): Get the Identifiers of Package Members
- [record](#): Get the data content of a specified data object
- [listRuns](#): Output a list of recorded runs to the console
- [viewRuns](#): Record relationships of objects in a DataPackage
- [deleteRuns](#): Record derivation relationships between objects in a DataPackage
- [publishRun](#): Upload all objects associated with a run to a repository
- [traceRuns](#): Trace processing lineage by finding related executions.
- [plotRuns](#): Trace processing lineage for a run and plot it.

See Also

[recordr](#) package description.

recordr_createObject *Provenance wrapper function for dataone::createObject method*

Description

Override the dataone::createObject method and record a provenance relationship for the object created.

Usage

```
recordr_createObject()
```

Note

This function is not intended to be called directly by a user.

recordr_getObject	<i>Provenance wrapper function for dataone::getObject</i>
-------------------	---

Description

Override the dataone::getObject method and record a provenance relationship for the object that was downloaded.

Usage

```
recordr_getObject()
```

Note

This function is not intended to be called directly by a user.

recordr_ggsave	<i>Provenance wrapper for the ggplot2::ggsave function</i>
----------------	--

Description

Override the ggplot2::ggsave function and record a provenance relationship for the file that was written.

Usage

```
recordr_ggsave()
```

Note

This function is not intended to be called directly by a user.

recordr_raster	<i>Provenance wrapper for the raster::raster function</i>
----------------	---

Description

Override the raster::raster function and record a provenance relationship for the file read.

Usage

```
recordr_raster()
```

Note

This function is not intended to be called directly by a user.

recordr_read.csv	<i>Provenance wrapper for the R utils::read.csv function</i>
------------------	--

Description

Override the utils::read.csv function and record a provenance relationship for the file that was read.

Usage

```
recordr_read.csv()
```

Arguments

... function parameters

Note

This function is not intended to be called directly by a user.

recordr_read_csv	<i>Provenance wrapper for the R readr::read_csv function</i>
------------------	--

Description

Override the readr::read_csv function and record a provenance relationship for the file that was read.

Usage

```
recordr_read_csv()
```

Arguments

... function parameters

Note

This function is not intended to be called directly by a user.

recordr_readLines	<i>Provenance wrapper for R base::readLines function</i>
-------------------	--

Description

Override the base::readLines function and record a provenance relationship for the file read.

Usage

```
recordr_readLines()
```

Note

This function is not intended to be called directly by a user.

recordr_readOGR	<i>Provenance wrapper for the rgdal::readOGR function</i>
-----------------	---

Description

Override the rgdal::readOGR function and record a provenance relationship for the file read.

Usage

```
recordr_readOGR()
```

Note

This function is not intended to be called directly by a user.

recordr_readPNG	<i>Provenance wrapper for the pnd::read function</i>
-----------------	--

Description

Override the png::read function and record a provenance relationship for the file read.

Usage

```
recordr_readPNG()
```

Note

This function is not intended to be called directly by a user.

recordr_scan	<i>Provenance wrapper for the R base::scan function</i>
--------------	---

Description

Override the base::scan function and record a provenance relationship for the scanned file.

Usage

```
recordr_scan()
```

Note

This function is not intended to be called directly by a user.

recordr_updateObject	<i>Provenance wrapper function for dataone::updateObject</i>
----------------------	--

Description

Override the dataone::updateObject method and record a provenance relationship for the object uploaded.

Usage

```
recordr_updateObject()
```

Note

This function is not intended to be called directly by a user.

recordr_write.csv	<i>Provenance wrapper for the R write.csv function</i>
-------------------	--

Description

Override the utils::write.csv function and record a provenance relationship for the written file.

Usage

```
recordr_write.csv()
```

Note

This function is not intended to be called directly by a user.

recordr_write_csv	<i>Provenance wrapper for the R readr::write_csv function</i>
-------------------	---

Description

Override the readr::write_csv function and record a provenance relationship for the written file.

Usage

```
recordr_write_csv()
```

Note

This function is not intended to be called directly by a user.

recordr_writeLines	<i>Provenance wrapper for R base::writeLines function</i>
--------------------	---

Description

Override the base::writeLines function and record a provenance relationship for the file that was written.

Usage

```
recordr_writeLines()
```

Note

This function is not intended to be called directly by a user.

recordr_writeOGR	<i>Provenance wrapper for the rgdal::writeOGR function</i>
------------------	--

Description

Override the rgdal::writeOGR function and record a provenance relationship for the file that was written.

Usage

```
recordr_writeOGR()
```

Note

This function is not intended to be called directly by a user.

recordr_writePNG	<i>Provenance wrapper for the png::write function</i>
------------------	---

Description

Override the png::write function and record a provenance relationship for the file that was written.

Usage

```
recordr_writePNG()
```

Note

This function is not intended to be called directly by a user.

recordr_writeRaster	<i>Provenance wrapper for the raster::writeRaster function</i>
---------------------	--

Description

Override the raster::writeRaster function and record a provenance relationship for the file that was written.

Usage

```
recordr_writeRaster()
```

Value

The name of the output file

Note

This function is not intended to be called directly by a user.

selectRuns	<i>Select runs that match search parameters</i>
------------	---

Description

This method is used to retrieve execution metadata for runs that match the search parameters.

Usage

```
selectRuns(recordr, ...)

## S4 method for signature 'Recordr'
selectRuns(recordr, runId = as.character(NA),
  script = as.character(NA), startTime = as.character(NA),
  endTime = as.character(NA), tag = as.character(NA),
  errorMessage = as.character(NA), seq = as.integer(NA),
  orderBy = "-startTime", delete = FALSE)
```

Arguments

recordr	A Recordr instance
...	additional parameters
runId	An execution identifiers
script	The file name of script to match.
startTime	Match executions that started after this time (inclusive)
endTime	Match executions that ended before this time (inclusive)
tag	The text of tag to match.
errorMessage	The text of error message to match.
seq	The run sequence number
orderBy	The column that will be used to sort the output. This can include a minus sign before the name, e.g. -startTime
delete	A logical value, if TRUE then the selected runs are deleted from the Recordr database.

Details

This method is used internally by the *recordr* package.

Value

A data.frame that contains execution metadata for executions that matched the search criteria

See Also

[Recordr](#) class description

standardizeCall	<i>Standardise a function call</i>
-----------------	------------------------------------

Description

Standardise a function call

Usage

```
standardizeCall(call, env = parent.frame())
```

Arguments

call	A call
env	Environment in which to look up call value.

Note

from Hadley Wicham's pryr standarize_call

startRecord	<i>Begin recording provenance for an R session.</i>
-------------	---

Description

This method starts the recording process and the method endRecord() completes it.

Usage

```
startRecord(recordr, ...)

## S4 method for signature 'Recordr'
startRecord(recordr, tag = as.character(NA),
  .file = as.character(NA), .console = TRUE, log = as.character(NA))
```

Arguments

recordr	a Recordr instance
...	additional parameters
tag	a string that is associated with this run
.file	the filename for the script to run (only used internally when startRecord() is called from record())
.console	a logical argument that is used internally by the recordr package
log	A character string. If .console=TRUE, the file to log console commands to. The default is 'console.log'.

Details

The `startRecord()` method can be called from the R console to begin a recording session during which provenance is captured for any functions that are inspected by Recordr. This recordr session can be closed by calling the `endRecord()` method. When the `record()` function is called to record a script, the `startRecord()` function is called automatically.

Value

execution identifier that uniquely identifies this recorded session

See Also

[Recordr](#) class description

Examples

```
## Not run:
rc <- new("Recordr")
startRecord(rc, tag="my first console run")
x <- read.csv(file="./test.csv")
runIdentifier <- endRecord(rc)

## End(Not run)
```

traceRuns

Trace processing lineage by finding related executions

Description

A data processing workflow might include multiple processing steps, with each step being performed by a separate R script. These multiple steps are linked by the files that one step writes and the next step in the workflow reads. The `traceRuns` method finds these connections between executions to determine the executions that comprise a processing workflow, and returns information for each run in the processing workflow including all files that were read and written by each script.

Usage

```
traceRuns(recordr, ...)

## S4 method for signature 'Recordr'
traceRuns(recordr, id = as.character(NA),
  file = as.character(NA), start = as.character(NA),
  end = as.character(NA), tag = as.character(NA),
  error = as.character(NA), seq = as.character(NA),
  orderBy = "-startTime", direction = "both", quiet = TRUE, ...)
```

Arguments

recordr	a Recordr instance
...	additional parameters
id	The identifier for a run. Either id or seq can be specified, not both.
file	The name of script to match
start	Match runs that started in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
end	Match runs that ended in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
tag	The text of tag to match
error	The text of error message to match.
seq	The sequence number for a run. #' @param id The execution identifier of a run to view
orderBy	Sort the results according to the specified column. A hyphen ('-') prepended to the column name denotes a descending sort. The default value is "-startTime"
direction	The direction to trace the lineage, either forward, backward, or both. The default is both
quiet	A logical if TRUE then output is not printed.

Details

If the run id or seq number is known for the run to be traced, then one or the other of these values can be used. Alternatively, other run attributes can be used to determine the run to be traced, such as file, start, etc. If these other search parameters are used and multiple runs are selected, only the first run selected will be traced. These search parameters can be used together to easily find certain runs, for example, the latest run of a particular script, the latest run with a specified tag specified, etc. (see examples).

Value

A list of the execution identifiers that are in the processing workflow.

See Also

[Recordr](#) class description

Examples

```
## Not run:
# Trace lineage for the run with sequence number '101'
linkedRuns <- traceRuns(recordr, seq=101)
# Trace lineage for the last execution of script "runModel.R"
linkedRuns <- traceRuns(recordr, file="runModel.R", orderBy="-startTime")
# Trace lineage for the last execution with the tag 'best run yet!' specified.
```

```
linkedRuns <- traceRuns(recordr, tag="best run yet!", orderBy="-startTime")

## End(Not run)
```

unArchiveFile	<i>Remove a file from the recordr archive directory</i>
---------------	---

Description

Remove a file from the recordr archive directory

Usage

```
unArchiveFile(recordr, fileId)
```

Arguments

recordr	A Recordr object
fileId	The fileId to remove from the archive

Value

A logical value - TRUE if the file is remove, FALSE if not

Note

This function is intended to run only during a record() session, i.e. the recordr environment needs to be available.

updateExecMeta	<i>Update a single execution metadata object.</i>
----------------	---

Description

Update an existing execution metadata entry with the values supplied.

Usage

```
updateExecMeta(recordr, ...)

## S4 method for signature 'Recordr'
updateExecMeta(recordr, executionId = as.character(NA),
  subject = as.character(NA), endTime = as.character(NA),
  errorMessage = as.character(NA), publishTime = as.character(NA),
  publishNodeId = as.character(NA), publishId = as.character(NA))
```

Arguments

recordr	A Recordr object
...	additional arguments
executionId	The execution id of the execution to be updated
subject	The authorized subject, i.e. from the client certificate.
endTime	The ending time of the execution.
errorMessage	An error message generated by the execution.
publishTime	The data and time that the execution was published
publishNodeId	The node identifier, e.g. "urn:node:testKNB" that the execution was published to.
publishId	The identifier that the execution was published with. In DataONE, this can be the identifier of the metadata object describing the datasets that were uploaded.

Details

Saved execution metadata is typically first stored when an execution begins, then updated at the end of a run (with error messages and ending time, for example). Also, execution can be updated when a run is published, with information about the publishing process.

See Also

[ExecMetadata](#) class description

upgradeRecordr	<i>Update the recordr database to the current version</i>
----------------	---

Description

Update the recordr database to the current version

Usage

```
upgradeRecordr(recordr)
```

Arguments

recordr	A recordr object
---------	------------------

Value

logical TRUE if the upgrade was successful, FALSE if a problem was encountered.

viewRuns	<i>View detailed information for an execution</i>
----------	---

Description

Detailed information for an execution is printed to the display.

Usage

```
viewRuns(recordr, ...)
```

```
## S4 method for signature 'Recordr'
viewRuns(recordr, id = as.character(NA),
  file = as.character(NA), start = as.character(NA),
  end = as.character(NA), tag = as.character(NA),
  error = as.character(NA), seq = as.character(NA),
  orderBy = "-startTime", sections = c("details", "used", "generated"),
  verbose = FALSE, page = TRUE, output = TRUE)
```

Arguments

recordr	A Recordr instance
...	additional parameter
id	The execution identifier of a run to view
file	The name of script to match
start	Match runs that started in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
end	Match runs that ended in this time range (inclusive) Times must be entered in the form 'YYYY-MM-DD HH:MM:SS' but can be shortened to not less than "YYYY"
tag	The text of tag to match
error	The text of error message to match.
seq	A run sequence number (can be a range, e.g seq=1:10)
orderBy	Sort the results according to the specified column. A hyphen ('-') prepended to the column name denotes a descending sort. The default value is "-startTime"
sections	Print the specified sections of the output. Default=c("details", "used", "generated")
verbose	a "logical", if TRUE then extra information is printed.
page	A logical value - if TRUE then pause after each run is displayed.
output	a "logical", if FALSE then no output is printed to the console (useful if only the returned object is needed).

Details

The execution and file information for runs that match the search criteria are printed to the console. The output is divided into three sections: "details", "used" and "generated". The "details" section shows execution information such as the start and end time of the run, run identifier, etc. The "used" section lists files that were read by a run. The "generated" section lists files that were created by a run. The list that is returned from "viewRuns" contains two elements - a data.frame with the execution information, and a data.frame that contains file information.

Value

A list that contains information about all selected runs.

See Also

[Recordr](#) class description

Examples

```
## Not run:
rc <- new("Recordr")
# View the tenth run that was recorded
viewRuns(rc, seq=10)
# View the first ten runs, with only the files "generated" section displayed
info <- viewRuns(rc, seq="1:10", sections="generated")
nrow(info$runs)
nrow(info$files)

## End(Not run)
```

writeExecMeta	<i>Save a single execution metadata.</i>
---------------	--

Description

Save a single execution metadata.

Usage

```
writeExecMeta(recordr, ...)

## S4 method for signature 'Recordr'
writeExecMeta(recordr, execMeta, ...)
```

Arguments

recordr	A Recordr object
...	Not yet used.
execMeta	an ExecMetadata object to save.

See Also

[ExecMetadata](#) class description

writeFileMeta	<i>Save metadata for a single file.</i>
---------------	---

Description

Metadata for a file is written to an RSQLite database.

Usage

```
writeFileMeta(recordr, fileMeta, ...)

## S4 method for signature 'Recordr,FileMetadata'
writeFileMeta(recordr, fileMeta, ...)
```

Arguments

recordr	A recordr object
fileMeta	A fileMetadata object
...	(Not yet used)

Details

This method is used internally by the recordr package.

See Also

[FileMetadata](#) class description

writeProvRel	<i>Save a single provenance relationship.</i>
--------------	---

Description

Metadata for a provenance relationship is written to the recordr RSQLite database.

Usage

```
writeProvRel(recordr, provRels, ...)

## S4 method for signature 'Recordr'
writeProvRel(recordr, provRels, ...)
```

Arguments

<code>recordr</code>	A recordr object
<code>provRels</code>	A ProvRels object.
<code>...</code>	(Not yet used)

Details

This method is used internally by the recordr package.

See Also

[ProvRels](#) class description

Index

changeHome, 3
coverageElement, 3

deleteRuns, 4, 25
deleteRuns, Recordr-method (deleteRuns), 4

endRecord, 5, 25
endRecord, Recordr-method (endRecord), 5
ExecMetadata, 10, 21, 37, 40
ExecMetadata-class, 5

FileMetadata, 11, 22, 40
FileMetadata-class, 7

geoCoverage, 8
getDBconnection, 8
getMetadata, 9
getMetadata, Recordr-method (getMetadata), 9

initialize, 6, 7, 18, 25
initialize, ExecMetadata-method, 9
initialize, FileMetadata-method, 11
initialize, ProvRels-method, 12
initialize, Recordr-method, 13
initialize-ExecMetadata (initialize, ExecMetadata-method), 9
initialize-FileMetadata (initialize, FileMetadata-method), 11
initialize-ProvRels (initialize, ProvRels-method), 12
initialize-Recordr (initialize, Recordr-method), 13

listRuns, 13, 25
listRuns, Recordr-method (listRuns), 13

makeEML, 15

plotRuns, 16, 25
plotRuns, Recordr-method (plotRuns), 16
ProvRels, 12, 23, 41
ProvRels-class, 17
publishRun, 18, 25
publishRun, Recordr-method (publishRun), 18
putMetadata, 19
putMetadata, Recordr-method (putMetadata), 19

readExecMeta, 6, 20
readExecMeta, Recordr-method (readExecMeta), 20
readFileMeta, 7, 21
readFileMeta, Recordr-method (readFileMeta), 21
readProvRels, 18, 22
readProvRels, Recordr-method (readProvRels), 22
record, 23, 25
record, Recordr-method (record), 23
Recordr, 4, 5, 9, 13, 14, 17–19, 23, 24, 32, 34, 35, 39
recordr, 6, 7, 18, 24, 25
Recordr-class, 24
recordr-package (recordr), 24
recordr_createObject, 25
recordr_getObject, 26
recordr_ggsave, 26
recordr_raster, 26
recordr_read.csv, 27
recordr_read_csv, 27
recordr_readLines, 28
recordr_readOGR, 28
recordr_readPNG, 28
recordr_scan, 29
recordr_updateObject, 29

- recordr_write.csv, [29](#)
- recordr_write_csv, [30](#)
- recordr_writeLines, [30](#)
- recordr_writeOGR, [30](#)
- recordr_writePNG, [31](#)
- recordr_writeRaster, [31](#)

- selectRuns, [32](#)
- selectRuns, Recordr-method (selectRuns),
 [32](#)
- standardizeCall, [33](#)
- startRecord, [25](#), [33](#)
- startRecord, Recordr-method
 (startRecord), [33](#)

- traceRuns, [25](#), [34](#)
- traceRuns, Recordr-method (traceRuns), [34](#)

- unArchiveFile, [36](#)
- updateExecMeta, [6](#), [36](#)
- updateExecMeta, Recordr-method
 (updateExecMeta), [36](#)
- upgradeRecordr, [37](#)

- viewRuns, [25](#), [38](#)
- viewRuns, Recordr-method (viewRuns), [38](#)

- writeExecMeta, [6](#), [39](#)
- writeExecMeta, Recordr-method
 (writeExecMeta), [39](#)
- writeFileMeta, [7](#), [40](#)
- writeFileMeta, Recordr, FileMetadata-method
 (writeFileMeta), [40](#)
- writeProvRel, [18](#), [40](#)
- writeProvRel, Recordr-method
 (writeProvRel), [40](#)