Package 'resourcer'

November 7, 2026

Type Package

Title Resource Resolver

Version 1.5.2

Description A resource represents some data or a computation unit. It is described by a URL and credentials. This package proposes a Resource model with ``resolver'' and ``client'' classes to facilitate the access and the usage of the resources.

License LGPL (>= 2.1)

Depends R (>= 3.5.0), R6, httr

Suggests testthat, knitr, haven, readr, readxl, ssh, sys, mongolite, dplyr, dbplyr, DBI, RMariaDB, RPostgres, sparklyr, nodbi, rmarkdown

URL https://www.obiba.org/resourcer/

BugReports https://github.com/obiba/resourcer/issues

VignetteBuilder knitr

Encoding UTF-8

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Yannick Marcon [aut, cre] (ORCID: <https://orcid.org/0000-0003-0138-2023>), OBiBa group [cph]

Maintainer Yannick Marcon <yannick.marcon@obiba.org>

Repository CRAN

Date/Publication 2026-11-07 13:10:02 UTC

Contents

as.data.frame.resource 3
as.data.frame.ResourceClient 3
as.resource.data.frame 4

as.resource.object . . . 4
as.resource.tbl . . . 5
CommandResourceClient . . . 5
DBIResourceConnector . . . 6
FileResourceClient . . . 8
FileResourceGetter . . . 9
findDBIResourceConnector . . . 10
findFileResourceGetter . . . 11
getDBIResourceConnectors . . . 11
getFileResourceGetters . . . 11
getResourceResolvers . . . 12
GridFsFileResourceGetter . . . 12
HttpFileResourceGetter . . . 13
LocalFileResourceGetter . . . 15
MariaDBResourceConnector . . . 16
newResource . . . 17
newResourceClient . . . 18
NoSQLResourceClient . . . 18
NoSQLResourceResolver . . . 20
OpalFileResourceGetter . . . 21
PostgresResourceConnector . . . 22
PrestoResourceConnector . . . 23
RDataFileResourceClient . . . 25
RDataFileResourceResolver . . . 26
RDSFileResourceClient . . . 27
RDSFileResourceResolver . . . 28
registerDBIResourceConnector . . . 29
registerFileResourceGetter . . . 30
registerResourceResolver . . . 30
resolveResource . . . 31
ResourceClient . . . 31
ResourceResolver . . . 33
ScpFileResourceGetter . . . 35
ShellResourceClient . . . 36
ShellResourceResolver . . . 37
SparkResourceConnector . . . 38
SQLResourceClient . . . 40
SQLResourceResolver . . . 41
SshResourceClient . . . 42
SshResourceResolver . . . 44
TidyFileResourceClient . . . 45
TidyFileResourceResolver . . . 46
unregisterDBIResourceConnector . . . 48
unregisterFileResourceGetter . . . 48
unregisterResourceResolver . . . 49

Index **50**

as.data.frame.resource

Coerce a resource to a data.frame

Description

Attempt to coerce a resource object to a data.frame: find a ResourceResolver and get the ResourceClient that will connect to the described dataset and make a data.frame of it.

Usage

```
## S3 method for class 'resource'
as.data.frame(x, ...)
```

Arguments

x	a resource object.
...	additional parameters, that may be used (or ignored) by the resource client.

Value

a data.frame (or a tibble)

as.data.frame.ResourceClient

Coerce a ResourceClient object to a data.frame

Description

Attempt to coerce a resource object to a data.frame: find a ResourceResolver and get the ResourceClient that will connect to the described dataset and make a data.frame of it.

Usage

```
## S3 method for class 'ResourceClient'
as.data.frame(x, ...)
```

Arguments

x	a ResourceClient object
...	additional parameters, that may be used (or ignored) by the resource client.

Value

a data.frame (or a tibble)

as.resource.data.frame *Coerce resource client to a data.frame*

Description

Coerce a ResourceClient object to a data.frame.

Usage

```
as.resource.data.frame(x, strict = FALSE, ...)
```

Arguments

x	The ResourceClient object to coerce to a data.frame.
strict	logical whether the resulting object must be strictly of class data.frame or if it can be a tibble.
...	Additional parameters, that may be used (or ignored) by the resource client.

Value

a data.frame (or a tibble)

as.resource.object *Coerce resource client to the internal data object*

Description

Coerce a ResourceClient object to internal data object: depending on the implementation of the ResourceClient, it can be a data connection object (like a DBI connection to a SQL database), or the actual data structure (when a resource is a R object extracted from a R data file for instance).

Usage

```
as.resource.object(x, ...)
```

Arguments

x	The ResourceClient object to coerce to a data.frame.
...	Additional parameters, that may be used (or ignored) by the resource client.

Value

the internal data object.

as.resource.tbl *Coerce resource client to a tbl*

Description

Coerce a ResourceClient object to a dplyr's tbl.

Usage

```
as.resource.tbl(x, ...)
```

Arguments

x	The ResourceClient object to coerce to a data.frame
...	Additional parameters, that may be used (or ignored) by the resource client.

Value

a dplyr's tbl

CommandResourceClient *Command resource client*

Description

Base class for resource clients issuing commands and get a result with the status of the execution, the output and the error messages.

Format

A R6 object of class CommandResourceClient

Super class

ResourceClient -> CommandResourceClient

Methods

Public methods:

- CommandResourceClient$new()
- CommandResourceClient$clone()

CommandResourceClient$new(): Creates a new CommandResourceClient instance

Usage:

```
CommandResourceClient$new(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A CommandResourceClient object.

`CommandResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
CommandResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`DBIResourceConnector` *DBI resource connector*

Description

Makes a DBI connection from a resource description, used in SQLResourceClient that is based on DBI.

Format

A R6 object of class DBIResourceConnector

Methods

Public methods:

- `DBIResourceConnector$new()`
- `DBIResourceConnector$isFor()`
- `DBIResourceConnector$createDBIConnection()`
- `DBIResourceConnector$getTableName()`
- `DBIResourceConnector$readDBTable()`
- `DBIResourceConnector$readDBTibble()`
- `DBIResourceConnector$closeDBIConnection()`
- `DBIResourceConnector$clone()`

`DBIResourceConnector$new()`: Creates a new DBIResourceConnector instance

Usage:

```
DBIResourceConnector$new()
```

Returns: A DBIResourceConnector object.

`DBIResourceConnector$isFor()`: Check that the provided parameter is of class "resource".

Usage:

```
DBIResourceConnector$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`DBIResourceConnector$createDBIConnection()`: Stub function which subclasses will implement to create a DBI connection object from a resource.

Usage:

```
DBIResourceConnector$createDBIConnection(resource)
```

Arguments:

`resource` A valid resource object.

`DBIResourceConnector$getTableName()`: Get the SQL table name from the resource URL.

Usage:

```
DBIResourceConnector$getTableName(resource)
```

Arguments:

`resource` A valid resource object.

Returns: The SQL table name.

`DBIResourceConnector$readDBTable()`: Read a table as a vanilla tibble using DBI connection object.

Usage:

```
DBIResourceConnector$readDBTable(conn, resource)
```

Arguments:

`conn` A DBI connection object.

`resource` A valid resource object.

Returns: A vanilla tibble.

`DBIResourceConnector$readDBTibble()`: Read a table as a SQL tibble using DBI connection object.

Usage:

```
DBIResourceConnector$readDBTibble(conn, resource)
```

Arguments:

`conn` A DBI connection object.

`resource` A valid resource object.

Returns: A SQL tibble.

`DBIResourceConnector$closeDBIConnection()`: Disconnect the DBI connection.

Usage:

```
DBIResourceConnector$closeDBIConnection(conn)
```

Arguments:

`conn` A DBI connection object.

`DBIResourceConnector$clone()`: The objects of this class are cloneable with this method.

Usage:

```
DBIResourceConnector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`FileResourceClient` *File resource client*

Description

Base class that connects to a file using a FileResourceGetter.

Format

A R6 object of class FileResourceClient

Super class

`ResourceClient` -> `FileResourceClient`

Methods

Public methods:

- `FileResourceClient$new()`
- `FileResourceClient$downloadFile()`
- `FileResourceClient$close()`
- `FileResourceClient$clone()`

`FileResourceClient$new()`: Creates a new FileResourceClient instance.

Usage:

```
FileResourceClient$new(resource, file.getter = NULL)
```

Arguments:

`resource` A valid resource object.

`file.getter` A FileResourceGetter object, optional. If not provided, it will be looked up in the FileResourceGetters registry. The operation will fail if none can be found.

Returns: A FileResourceClient object.

`FileResourceClient$downloadFile()`: Performs the file download, if it does not already exists locally.

Usage:

```
FileResourceClient$downloadFile()
```

Returns: The local path to the downloaded file.

`FileResourceClient$close()`: Removes the file if it was downloaded. A local file resource will remain untouched.

Usage:

```
FileResourceClient$close()
```

`FileResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
FileResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`FileResourceGetter` *File resource getter*

Description

Helper base class for getting the file described by the resource object. ResourceClient class implementations can use this utility to retrieve files from any locations before processing them according to the declared data format.

Format

A R6 object of class FileResourceGetter

Methods

Public methods:

- `FileResourceGetter$new()`
- `FileResourceGetter$isFor()`
- `FileResourceGetter$downloadFile()`
- `FileResourceGetter$extractFileName()`
- `FileResourceGetter$makeDownloadDir()`
- `FileResourceGetter$clone()`

`FileResourceGetter$new()`**:** Creates a new FileResourceGetter instance.

Usage:

```
FileResourceGetter$new()
```

Returns: A FileResourceGetter object.

`FileResourceGetter$isFor()`**:** Check that the provided parameter is of class "resource" and has a format defined.

Usage:

```
FileResourceGetter$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`FileResourceGetter$downloadFile()`**:** Stub function which subclasses will implement to make a "resource.file" object from a resource.

Usage:

```
FileResourceGetter$downloadFile(resource, ...)
```

Arguments:

`resource` A valid resource object.

`...` Additional parameters that may be relevant for FileResourceGetter subclasses.

Returns: A "resource.file" object.

`FileResourceGetter$extractFileName()`: Utility to get the base name from the file path.

Usage:

```
FileResourceGetter$extractFileName(resource)
```

Arguments:

`resource` A valid resource object.

Returns: The file base name.

`FileResourceGetter$makeDownloadDir()`: Creates a directory where to download file in the R session's temporary directory. This directory will be flushed when the R session ends.

Usage:

```
FileResourceGetter$makeDownloadDir()
```

Returns: The path to the download directory.

`FileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
FileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`findDBIResourceConnector` *Find a DBI resource connector*

Description

Find the DBI resource connector that will download the DBI from the provided resource object.

Usage

```
findDBIResourceConnector(x)
```

Arguments

`x`	The resource object which corresponding DBI connector is to be found.

Value

The corresponding DBIResourceConnector object or NULL if none applies.

findFileResourceGetter *Find a file resource getter*

Description

Find the file resource getter that will download the file from the provided resource object.

Usage

```
findFileResourceGetter(x)
```

Arguments

x	The resource object which corresponding file getter is to be found.

Value

The corresponding FileResourceGetter object or NULL if none applies.

getDBIResourceConnectors *Get DBI resource connectors registry*

Description

Get the DBIResourceConnectors registry, create it if it does not exist.

Usage

```
getDBIResourceConnectors()
```

getFileResourceGetters *Get file resource getters registry*

Description

Get the FileResourceGetters registry, create it if it does not exist.

Usage

```
getFileResourceGetters()
```

getResourceResolvers *Get resource resolvers registry*

Description

Get the resource resolvers registry, create it if it does not exist.

Usage

```
getResourceResolvers()
```

Examples

```
{
resourcer::getResourceResolvers()
}
```

GridFsFileResourceGetter *GridFS file resource getter*

Description

Access a file that is in the MongoDB file store (GridFS). Credentials may apply.

Format

A R6 object of class GridFsFileResourceGetter

Super class

`FileResourceGetter` -> `GridFsFileResourceGetter`

Methods

Public methods:

- `GridFsFileResourceGetter$new()`
- `GridFsFileResourceGetter$isFor()`
- `GridFsFileResourceGetter$downloadFile()`
- `GridFsFileResourceGetter$clone()`

`GridFsFileResourceGetter$new()`: Creates a new GridFsFileResourceGetter instance.

Usage:

```
GridFsFileResourceGetter$new()
```

Returns: A GridFsFileResourceGetter object.

`GridFsFileResourceGetter$isFor()`: Check that the provided resource has a URL that locates a GridFS object: either the URL scheme is "gridfs" or it is "mongodb"/"mongodb+srv" with a query parameter "prefix=fs" (that identifies GridFS collection names).

Usage:

```
GridFsFileResourceGetter$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`GridFsFileResourceGetter$downloadFile()`: Download the file from the MongoDB file store in a temporary location.

Usage:

```
GridFsFileResourceGetter$downloadFile(resource, ...)
```

Arguments:

`resource` A valid resource object.

`...` Unused additional parameters.

Returns: The "resource.file" object.

`GridFsFileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
GridFsFileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`HttpFileResourceGetter`

HTTP file resource getter

Description

Access a file that is stored at a HTTP(S) address. Use Basic authentication header if both resource's identity and secret are defined.

Format

A R6 object of class HttpFileResourceGetter

Super class

`FileResourceGetter` -> `HttpFileResourceGetter`

Methods

Public methods:

- HttpFileResourceGetter$new()
- HttpFileResourceGetter$isFor()
- HttpFileResourceGetter$downloadFile()
- HttpFileResourceGetter$clone()

`HttpFileResourceGetter$new()`: Creates a new HttpFileResourceGetter instance.

Usage:

```
HttpFileResourceGetter$new()
```

Returns: A HttpFileResourceGetter object.

`HttpFileResourceGetter$isFor()`: Check that the provided resource has a URL that locates a file accessible through "http" or "https".

Usage:

```
HttpFileResourceGetter$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`HttpFileResourceGetter$downloadFile()`: Download the file from the remote address in a temporary location. Applies Basic authentication if credentials are provided in the resource.

Usage:

```
HttpFileResourceGetter$downloadFile(resource, ...)
```

Arguments:

`resource` A valid resource object.

`...` Unused additional parameters.

Returns: The "resource.file" object.

`HttpFileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
HttpFileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

LocalFileResourceGetter *Local file resource getter*

Description

Access a file that is stored in the local file system. No credentials apply.

Format

A R6 object of class LocalFileResourceGetter

Super class

`FileResourceGetter` -> `LocalFileResourceGetter`

Methods

Public methods:

- `LocalFileResourceGetter$new()`
- `LocalFileResourceGetter$isFor()`
- `LocalFileResourceGetter$downloadFile()`
- `LocalFileResourceGetter$clone()`

`LocalFileResourceGetter$new()`: Creates a new LocalFileResourceGetter instance.

Usage:

```
LocalFileResourceGetter$new()
```

Returns: A LocalFileResourceGetter object.

`LocalFileResourceGetter$isFor()`: Check that the provided resource has a URL that locates a file stored in the local file system.

Usage:

```
LocalFileResourceGetter$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`LocalFileResourceGetter$downloadFile()`: Make a "resource.file" object from a local file resource.

Usage:

```
LocalFileResourceGetter$downloadFile(resource, ...)
```

Arguments:

`resource` A valid resource object.

... Unused additional parameters.

Returns: The "resource.file" object.

`LocalFileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
LocalFileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`MariaDBResourceConnector`

MariaDB DBI resource connector

Description

Makes a MariaDB/MySQL DBI connection from a resource description.

Format

A R6 object of class MariaDBResourceConnector

Super class

`DBIResourceConnector` -> `MariaDBResourceConnector`

Methods

Public methods:

- `MariaDBResourceConnector$new()`
- `MariaDBResourceConnector$isFor()`
- `MariaDBResourceConnector$createDBIConnection()`
- `MariaDBResourceConnector$clone()`

`MariaDBResourceConnector$new()`: Creates a new MariaDBResourceConnector instance.

Usage:

```
MariaDBResourceConnector$new()
```

Returns: A MariaDBResourceConnector object.

`MariaDBResourceConnector$isFor()`: Check that the provided resource has a URL that locates a MySQL or MariaDB object: the URL scheme must be "mysql" or "mariadb".

Usage:

```
MariaDBResourceConnector$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`MariaDBResourceConnector$createDBIConnection()`: Creates a DBI connection object from a resource.

Usage:

```
MariaDBResourceConnector$createDBIConnection(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A DBI connection object.

`MariaDBResourceConnector$clone()`: The objects of this class are cloneable with this method.

Usage:

```
MariaDBResourceConnector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`newResource` *Create a Resource*

Description

Creates a new Resource structure.

Usage

```
newResource(name = "", url, identity = NULL, secret = NULL, format = NULL)
```

Arguments

`name`	Otpional human friendly name that identifies the resource.
`url`	URL to access the resource whether it is data or computation capability.
`identity`	User name or account ID (if credentials are applicable).
`secret`	User password or token (if credentials are applicable).
`format`	Data format, to help resource resolver identification and coercing to other formats, optional.

Examples

```
{
# make a SPSS file resource
res <- resourcer::newResource(
  name = "CNSIM1",
  url = "file:///data/CNSIM1.sav",
  format = "spss"
)
}
```

newResourceClient *Creates a resource client*

Description

From a resource object, find the corresponding resolver in the resolver registry and create a new resource client.

Usage

```
newResourceClient(x)
```

Arguments

x	The resource object which corresponding resolver is to be found.

Value

The corresponding ResourceClient object or NULL if none applies.

Examples

```
library(resourcer)
res <- newResource(
  name = "CNSIM1",
  url = "file:///data/CNSIM1.sav",
  format = "spss"
)
client <- newResourceClient(res)
```

NoSQLResourceClient *NoSQL database resource client*

Description

Resource client that connects to a NoSQL database supported by nodbi.

Format

A R6 object of class NoSQLResourceClient

Super class

ResourceClient -> NoSQLResourceClient

Methods

Public methods:

- NoSQLResourceClient$new()
- NoSQLResourceClient$getConnection()
- NoSQLResourceClient$asDataFrame()
- NoSQLResourceClient$getDatabaseName()
- NoSQLResourceClient$getTableName()
- NoSQLResourceClient$close()
- NoSQLResourceClient$clone()

`NoSQLResourceClient$new()`: Creates a new NoSQLResourceClient instance.

Usage:

```
NoSQLResourceClient$new(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A NoSQLResourceClient object.

`NoSQLResourceClient$getConnection()`: Creates the nodbi connection object if it does not exist.

Usage:

```
NoSQLResourceClient$getConnection()
```

Returns: The nodbi connection object.

`NoSQLResourceClient$asDataFrame()`: Makes a data.frame from the remote database table.

Usage:

```
NoSQLResourceClient$asDataFrame()
```

Returns: A tibble.

`NoSQLResourceClient$getDatabaseName()`: Extract the database name from the resource URL.

Usage:

```
NoSQLResourceClient$getDatabaseName()
```

Returns: The database name.

`NoSQLResourceClient$getTableName()`: Extract the database table name from the resource URL.

Usage:

```
NoSQLResourceClient$getTableName()
```

Returns: The database table name.

`NoSQLResourceClient$close()`: Close the nodbi connection.

Usage:

```
NoSQLResourceClient$close()
```

`NoSQLResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
NoSQLResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`NoSQLResourceResolver` *NoSQL Database Resource resolver*

Description

The resource is NoSQL database such as mongodb (elasticsearch, redis, couchdb, sqlite are not supported yet).

Format

A R6 object of class NoSQLResourceResolver

Super class

`ResourceResolver` -> `NoSQLResourceResolver`

Methods

Public methods:

- `NoSQLResourceResolver$isFor()`
- `NoSQLResourceResolver$newClient()`
- `NoSQLResourceResolver$clone()`

`NoSQLResourceResolver$isFor()`: Check that the provided resource has a URL that locates a nodbi object: the URL scheme must be one of "mongodb", "mongodb+srv". Other NoSQL databases "elasticsearch", "redis", "couchdb", "sqlite" are not supported yet.

Usage:

```
NoSQLResourceResolver$isFor(x)
```

Arguments:

`x` The resource object to validate.

Returns: A logical.

`NoSQLResourceResolver$newClient()`: Creates a NoSQLResourceClient instance from provided resource.

Usage:

```
NoSQLResourceResolver$newClient(x)
```

Arguments:

x A valid resource object.

Returns: A NoSQLResourceClient object.

`NoSQLResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
NoSQLResourceResolver$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

`OpalFileResourceGetter`

Opal file resource getter

Description

Access a file that is stored in a Opal server. Use Basic authentication header if both resource's identity and secret are defined, token authentication if secret only is defined.

Format

A R6 object of class OpalFileResourceGetter

Super class

`FileResourceGetter` -> `OpalFileResourceGetter`

Methods

Public methods:

- `OpalFileResourceGetter$new()`
- `OpalFileResourceGetter$isFor()`
- `OpalFileResourceGetter$downloadFile()`
- `OpalFileResourceGetter$clone()`

`OpalFileResourceGetter$new()`: Creates a new OpalFileResourceGetter instance.

Usage:

```
OpalFileResourceGetter$new()
```

Returns: A OpalFileResourceGetter object.

`OpalFileResourceGetter$isFor()`: Check that the provided resource has a URL that locates a Opal file: the URL scheme must be "opal+http" or "opal+https" and the path must designate a file web service entry point (i.e. starts with "ws/files/").

Usage:

```
OpalFileResourceGetter$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`OpalFileResourceGetter$downloadFile()`: Download the file from the Opal file system in a temporary location.

Usage:

```
OpalFileResourceGetter$downloadFile(resource, ...)
```

Arguments:

`resource` A valid resource object.

`...` Unused additional parameters.

Returns: The "resource.file" object.

`OpalFileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
OpalFileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`PostgresResourceConnector`

Postgres DBI resource connector

Description

Makes a Postgres DBI connection from a resource description.

Format

A R6 object of class PostgresResourceConnector

Super class

`DBIResourceConnector` -> `PostgresResourceConnector`

Methods

Public methods:

- `PostgresResourceConnector$new()`
- `PostgresResourceConnector$isFor()`
- `PostgresResourceConnector$createDBIConnection()`
- `PostgresResourceConnector$clone()`

`PostgresResourceConnector$new()`: Creates a new PostgresResourceConnector instance.

Usage:

```
PostgresResourceConnector$new()
```

Returns: A PostgresResourceConnector object.

`PostgresResourceConnector$isFor()`: Check that the provided resource has a URL that locates a Postgres object: the URL scheme must be "postgres" or "postgresql".

Usage:

```
PostgresResourceConnector$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`PostgresResourceConnector$createDBIConnection()`: Creates a DBI connection object from a resource.

Usage:

```
PostgresResourceConnector$createDBIConnection(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A DBI connection object.

`PostgresResourceConnector$clone()`: The objects of this class are cloneable with this method.

Usage:

```
PostgresResourceConnector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`PrestoResourceConnector`

Presto DBI resource connector

Description

Makes a Presto DBI connection from a resource description.

Format

A R6 object of class PrestoResourceConnector

Super class

`DBIResourceConnector` -> `PrestoResourceConnector`

Methods

Public methods:

- PrestoResourceConnector$new()
- PrestoResourceConnector$isFor()
- PrestoResourceConnector$createDBIConnection()
- PrestoResourceConnector$clone()

`PrestoResourceConnector$new()`: Creates a new PrestoResourceConnector instance.

Usage:

```
PrestoResourceConnector$new()
```

Returns: A PrestoResourceConnector object.

`PrestoResourceConnector$isFor()`: Check that the provided resource has a URL that locates a Presto object: the URL scheme must be "presto", "presto+http" or "presto+https".

Usage:

```
PrestoResourceConnector$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

`PrestoResourceConnector$createDBIConnection()`: Creates a DBI connection object from a resource.

Usage:

```
PrestoResourceConnector$createDBIConnection(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A DBI connection object.

`PrestoResourceConnector$clone()`: The objects of this class are cloneable with this method.

Usage:

```
PrestoResourceConnector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

RDataFileResourceClient

R data file resource client

Description

Connects to a R data file and loads it in a controlled environment.

Format

A R6 object of class RDSFileResourceClient

Super classes

ResourceClient -> FileResourceClient -> RDataFileResourceClient

Methods

Public methods:

- RDataFileResourceClient$new()
- RDataFileResourceClient$asDataFrame()
- RDataFileResourceClient$getValue()
- RDataFileResourceClient$clone()

RDataFileResourceClient$new(): Creates a new RDataFileResourceClient instance.

Usage:

```
RDataFileResourceClient$new(resource)
```

Arguments:

resource A valid resource object.

Returns: A RDataFileResourceClient object.

RDataFileResourceClient$asDataFrame(): Coerce the resource value extracted from the R data file to a data.frame.

Usage:

```
RDataFileResourceClient$asDataFrame(...)
```

Arguments:

... Additional parameters to as.data.frame (not used yet).

Returns: A data.frame.

RDataFileResourceClient$getValue(): Get the resource value extracted from the R data file.

Usage:

```
RDataFileResourceClient$getValue(...)
```

Arguments:

... Additional parameters to get the value object (not used yet).

Returns: The resource value.

`RDataFileResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
RDataFileResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`RDataFileResourceResolver`

R data file Resource resolver

Description

The resource is a R data file and data format is the class of the symbol that will be loaded.

Format

A R6 object of class RDataFileResourceResolver

Super class

`ResourceResolver` -> `RDataFileResourceResolver`

Methods

Public methods:

- `RDataFileResourceResolver$isFor()`
- `RDataFileResourceResolver$newClient()`
- `RDataFileResourceResolver$clone()`

`RDataFileResourceResolver$isFor()`: Check that the provided resource has a URL that locates a R data file: the resource can be accessed as a file and the resource URL path ends with ".rda" or ".rdata" (case ignored), or the resource format is prefixed with "r:" or "rda:" (a kind of namespace to qualify the R object class).

Usage:

```
RDataFileResourceResolver$isFor(x)
```

Arguments:

`x` The resource object to validate.

Returns: A logical.

`RDataFileResourceResolver$newClient()`: Creates a RDataFileResourceClient instance from provided resource.

Usage:

```
RDataFileResourceResolver$newClient(x)
```

Arguments:

`x` A valid resource object.

Returns: A RDataFileResourceClient object.

`RDataFileResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
RDataFileResourceResolver$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`RDSFileResourceClient` *R object file resource client*

Description

Connects to a RDS file and loads the serialized object. Similar to the R data file resource, except that the RDS format stores a single R object.

Format

A R6 object of class RDSFileResourceClient

Super classes

`ResourceClient` -> `FileResourceClient` -> `RDSFileResourceClient`

Methods

Public methods:

- `RDSFileResourceClient$new()`
- `RDSFileResourceClient$asDataFrame()`
- `RDSFileResourceClient$getValue()`
- `RDSFileResourceClient$clone()`

`RDSFileResourceClient$new()`: Creates a new RDSFileResourceClient instance.

Usage:

```
RDSFileResourceClient$new(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A RDSFileResourceClient object.

`RDSFileResourceClient$asDataFrame()`: Coerce the resource value extracted from the R object file to a data.frame.

Usage:

```
RDSFileResourceClient$asDataFrame(...)
```

Arguments:

`...` Additional parameters to as.data.frame (not used yet).

Returns: A data.frame.

`RDSFileResourceClient$getValue()`: Get the resource value extracted from the R object file.

Usage:

```
RDSFileResourceClient$getValue(...)
```

Arguments:

`...` Additional parameters to get the value object (not used yet).

Returns: The resource value.

`RDSFileResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
RDSFileResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`RDSFileResourceResolver`

R object file Resource resolver

Description

The resource is a RDS file.

Format

A R6 object of class RDSFileResourceResolver

Super class

`ResourceResolver` -> `RDSFileResourceResolver`

Methods

Public methods:

- RDSFileResourceResolver$isFor()
- RDSFileResourceResolver$newClient()
- RDSFileResourceResolver$clone()

`RDSFileResourceResolver$isFor()`: Check that the provided resource has a URL that locates a R object file: the resource can be accessed as a file and the resource URL path ends with ".rds" (case ignored), or the resource format is prefixed with "rds:" (a kind of namespace to qualify the R object class).

Usage:

```
RDSFileResourceResolver$isFor(x)
```

Arguments:

`x` The resource object to validate.

Returns: A logical.

`RDSFileResourceResolver$newClient()`: Creates a RDSFileResourceClient instance from provided resource.

Usage:

```
RDSFileResourceResolver$newClient(x)
```

Arguments:

`x` A valid resource object.

Returns: A RDSFileResourceClient object.

`RDSFileResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
RDSFileResourceResolver$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`registerDBIResourceConnector` *Register a DBI resource connector*

Description

Maintain an list of `DBIResourceConnectors` that will be called when a new DBI resource connector is to be found.

Usage

```
registerDBIResourceConnector(x)
```

Arguments

`x`	The DBI resource connector object to register.

registerFileResourceGetter

Register a file resource getter

Description

Maintain an list of `FileResourceGetter`s that will be called when a new file resource getter is to be found.

Usage

```
registerFileResourceGetter(x)
```

Arguments

`x`	The file resource getter object to register.

registerResourceResolver

Register a resource resolver

Description

Maintain an list of resource resolvers that will be called when a new resource is to be resolved.

Usage

```
registerResourceResolver(x)
```

Arguments

`x`	The resource resolver object to register.

Examples

```
## Not run:
resourcer::registerResourceResolver(MyFileFormatResourceResolver$new())

## End(Not run)
```

resolveResource *Find a resource resolver*

Description

Find the resolver that will make a client from the provided resource object.

Usage

```
resolveResource(x)
```

Arguments

x	The resource object which corresponding resolver is to be found.

Value

The corresponding ResourceResolver object or NULL if none applies.

Examples

```
library(resourcer)
res <- newResource(
  name = "CNSIM1",
  url = "file:///data/CNSIM1.sav",
  format = "spss"
)
resolver <- resolveResource(res)
```

ResourceClient *Resource client*

Description

Helper class for connecting to a resource data store or a computation unit.

Format

A R6 object of class ResourceClient

Methods

Public methods:

- ResourceClient$new()
- ResourceClient$getResource()
- ResourceClient$getConnection()
- ResourceClient$downloadFile()
- ResourceClient$asDataFrame()
- ResourceClient$asTbl()
- ResourceClient$exec()
- ResourceClient$close()
- ResourceClient$clone()

`ResourceClient$new()`: Creates a ResourceClient instance.

Usage:

```
ResourceClient$new(resource)
```

Arguments:

`resource` The resource object to be interprated.

Returns: A ResourceClient object.

`ResourceClient$getResource()`: Get the resource object.

Usage:

```
ResourceClient$getResource()
```

Returns: The resource object.

`ResourceClient$getConnection()`: Get the implementation-specific object that connects to the resource

Usage:

```
ResourceClient$getConnection()
```

Returns: The connection object.

`ResourceClient$downloadFile()`: Stub function to be implemented by subclasses if relevant. Get the resource as a local file.

Usage:

```
ResourceClient$downloadFile(...)
```

Arguments:

`...` Additional parameters.

Returns: The path to the local file.

`ResourceClient$asDataFrame()`: Stub function to be implemented by subclasses if relevant. Coerce the resource as a data.frame.

Usage:

```
ResourceClient$asDataFrame(...)
```

Arguments:

... Additional parameters.

Returns: A data.frame object (can also be a tibble).

`ResourceClient$asTbl()`: Stub function to be implemented by subclasses if relevant. Coerce the resource as a dplyr's tbl.

Usage:

```
ResourceClient$asTbl(...)
```

Arguments:

... Additional parameters.

Returns: A dplyr's tbl object.

`ResourceClient$exec()`: Stub function to be implemented by subclasses if relevant. Executes a command on a computation resource.

Usage:

```
ResourceClient$exec(...)
```

Arguments:

... Additional parameters that will represent the command to execute.

Returns: A command execution result object.

`ResourceClient$close()`: Silently closes the connection to the resource

Usage:

```
ResourceClient$close()
```

`ResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`ResourceResolver` *Resource resolver*

Description

Helper class for building a Client that implements the access to the data or the computation unit.

Format

A R6 object of class ResourceResolver

Methods

Public methods:

- ResourceResolver$new()
- ResourceResolver$isFor()
- ResourceResolver$newClient()
- ResourceResolver$clone()

`ResourceResolver$new()`: Creates a new ResourceResolver instance.

Usage:

```
ResourceResolver$new()
```

Returns: A ResourceResolver object.

`ResourceResolver$isFor()`: Check that the provided object is of class "resource".

Usage:

```
ResourceResolver$isFor(x)
```

Arguments:

`x` The resource object to evaluate.

Returns: A logical.

`ResourceResolver$newClient()`: Stub function to be implemented by subclasses. Makes an object which class inherits from ResourceClient.

Usage:

```
ResourceResolver$newClient(x)
```

Arguments:

`x` The resource object to evaluate.

Returns: The ResourceClient object that will access the provided resource.

`ResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ResourceResolver$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

ScpFileResourceGetter *SCP file resource getter*

Description

Access a file that is stored on a server accessible through SSH. Credentials apply.

Format

A R6 object of class ScpFileResourceGetter

Super class

FileResourceGetter -> ScpFileResourceGetter

Methods

Public methods:

- ScpFileResourceGetter$new()
- ScpFileResourceGetter$isFor()
- ScpFileResourceGetter$downloadFile()
- ScpFileResourceGetter$clone()

ScpFileResourceGetter$new(): Creates a ScpFileResourceGetter instance.

Usage:

```
ScpFileResourceGetter$new()
```

Returns: The ScpFileResourceGetter object.

ScpFileResourceGetter$isFor(): Check that the provided resource is a file accessible by SCP: the resource URL scheme must be "scp".

Usage:

```
ScpFileResourceGetter$isFor(resource)
```

Arguments:

resource The resource object to evaluate.

Returns: A logical.

ScpFileResourceGetter$downloadFile(): Download the file described by the resource over an SSH connection.

Usage:

```
ScpFileResourceGetter$downloadFile(resource, ...)
```

Arguments:

resource The resource that identifies the file.

... Additional parameters (not used).

Returns: The "resource.file" object.

`ScpFileResourceGetter$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ScpFileResourceGetter$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`ShellResourceClient` *Shell resource client*

Description

Executes local system shell commands.

Format

A R6 object of class ShellResourceClient

Super classes

`ResourceClient` -> `CommandResourceClient` -> `ShellResourceClient`

Methods

Public methods:

- `ShellResourceClient$new()`
- `ShellResourceClient$getAllowedCommands()`
- `ShellResourceClient$copyFile()`
- `ShellResourceClient$exec()`
- `ShellResourceClient$clone()`

`ShellResourceClient$new()`: Create a ShellResourceClient instance. This client will interact wtih a computation resource using shell commands.

Usage:

```
ShellResourceClient$new(resource)
```

Arguments:

`resource` The computation resource.

Returns: The ShellResourceClient object.

`ShellResourceClient$getAllowedCommands()`: Get the command names that can be executed.

Usage:

```
ShellResourceClient$getAllowedCommands()
```

Returns: A character vector

`ShellResourceClient$copyFile()`: Copy one or more files (wilcard * is supported in the file name (which can be a directory))

Usage:

```
ShellResourceClient$copyFile(file, to = ".", verbose = FALSE)
```

Arguments:

`file` The file to copy.

`to` The copy destination.

`verbose` If TRUE, details the file operations on the console output.

Returns: The path to the files having been copied.

`ShellResourceClient$exec()`: Executes a shell command in the working directory specified in the resource's URL.

Usage:

```
ShellResourceClient$exec(command, params = NULL, test = FALSE)
```

Arguments:

`command` The command name.

`params` A character vector of arguments to pass.

`test` If TRUE, the command is printed but not executed (for debugging).

Returns: The command execution result object.

`ShellResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ShellResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`ShellResourceResolver` *Shell Resource resolver*

Description

The resource is a computation unit, accessible by issuing local system commands, i.e. which URL scheme is "sh".

Format

A R6 object of class ShellResourceResolver

Super class

`ResourceResolver` -> `ShellResourceResolver`

Methods

Public methods:

- ShellResourceResolver$isFor()
- ShellResourceResolver$newClient()
- ShellResourceResolver$clone()

`ShellResourceResolver$isFor()`: Check that the provided resource is a computation resource accessible by shell commands. The resource URL scheme must be "sh" or "shell".

Usage:

```
ShellResourceResolver$isFor(x)
```

Arguments:

`x` The resource object.

Returns: A logical.

`ShellResourceResolver$newClient()`: Create a ShellResourceClient instance from the provided resource.

Usage:

```
ShellResourceResolver$newClient(x)
```

Arguments:

`x` A valid resource object.

Returns: A ShellResourceClient object.

`ShellResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ShellResourceResolver$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`SparkResourceConnector`

Apache Spark DBI resource connector

Description

Makes a Apache Spark connection object, that is also a DBI connection object, from a resource description.

Format

A R6 object of class SparkResourceConnector

Super class

DBIResourceConnector -> SparkResourceConnector

Methods

Public methods:

- SparkResourceConnector$new()
- SparkResourceConnector$isFor()
- SparkResourceConnector$createDBIConnection()
- SparkResourceConnector$closeDBIConnection()
- SparkResourceConnector$clone()

SparkResourceConnector$new(): Create a SparkResourceConnector instance.

Usage:

```
SparkResourceConnector$new()
```

Returns: A SparkResourceConnector object.

SparkResourceConnector$isFor(): Check if the provided resource applies to a Apache Spark server. The resource URL scheme must be one of "spark", "spark+http" or "spark+https".

Usage:

```
SparkResourceConnector$isFor(resource)
```

Arguments:

`resource` The resource object to validate.

Returns: A logical.

SparkResourceConnector$createDBIConnection(): Creates a DBI connection object from a Apache Spark resource.

Usage:

```
SparkResourceConnector$createDBIConnection(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A DBI connection object.

SparkResourceConnector$closeDBIConnection(): Close the DBI connection to Apache Spark.

Usage:

```
SparkResourceConnector$closeDBIConnection(conn)
```

Arguments:

`conn` A DBI connection object.

SparkResourceConnector$clone(): The objects of this class are cloneable with this method.

Usage:

```
SparkResourceConnector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

SQLResourceClient *SQL database resource client*

Description

Resource client that connects to a SQL database supported by DBI.

Format

A R6 object of class SQLResourceClient

Super class

ResourceClient -> SQLResourceClient

Methods

Public methods:

- SQLResourceClient$new()
- SQLResourceClient$getConnection()
- SQLResourceClient$asDataFrame()
- SQLResourceClient$asTbl()
- SQLResourceClient$close()
- SQLResourceClient$clone()

SQLResourceClient$new(): Creates a SQLResourceClient from a resource.

Usage:

```
SQLResourceClient$new(resource, dbi.connector = NULL)
```

Arguments:

`resource` The resource object.

`dbi.connector` An optional DBIResourceConnector object. If not provided, it will be looked up in the DBIResourceConnector registry.

Returns: The SQLResourceClient object.

SQLResourceClient$getConnection(): Get or create the DBI connection object that will access the resource.

Usage:

```
SQLResourceClient$getConnection()
```

Returns: The DBI connection object.

SQLResourceClient$asDataFrame(): Coerce the SQL table to a data.frame.

Usage:

```
SQLResourceClient$asDataFrame(...)
```

Arguments:

... Additional parameters (not used).

Returns: A data.frame (more specifically a tibble).

`SQLResourceClient$asTbl()`: Get the SQL table as a dplyr's tbl.

Usage:

```
SQLResourceClient$asTbl()
```

Returns: A dplyr's tbl object.

`SQLResourceClient$close()`: Silently close the DBI connection.

Usage:

```
SQLResourceClient$close()
```

`SQLResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
SQLResourceClient$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

`SQLResourceResolver` *SQL Database Resource resolver*

Description

The resource is SQL database.

Format

A R6 object of class SQLResourceResolver

Super class

`ResourceResolver` -> `SQLResourceResolver`

Methods

Public methods:

- `SQLResourceResolver$isFor()`
- `SQLResourceResolver$newClient()`
- `SQLResourceResolver$clone()`

`SQLResourceResolver$isFor()`: Check that the provided resource has a registered DBIResourceConnector.

Usage:

`SQLResourceResolver$isFor(x)`

Arguments:

`x` The resource object to evaluate.

Returns: A logical.

`SQLResourceResolver$newClient()`: Creates a SQLResourceClient instance from provided resource.

Usage:

`SQLResourceResolver$newClient(x)`

Arguments:

`x` A valid resource object.

Returns: A SQLResourceClient object.

`SQLResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

`SQLResourceResolver$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`SshResourceClient` *SSH resource client*

Description

Connects to a SSH server.

Format

A R6 object of class SshResourceClient

Super classes

`ResourceClient` -> `CommandResourceClient` -> `SshResourceClient`

Methods

Public methods:

- `SshResourceClient$new()`
- `SshResourceClient$getAllowedCommands()`
- `SshResourceClient$getConnection()`
- `SshResourceClient$downloadFile()`
- `SshResourceClient$uploadFile()`
- `SshResourceClient$tempDir()`

- SshResourceClient$removeTempDir()
- SshResourceClient$exec()
- SshResourceClient$close()
- SshResourceClient$clone()

`SshResourceClient$new()`: Create a SshResourceClient instance. This client will interact wtih a computation resource using ssh commands.

Usage:

```
SshResourceClient$new(resource)
```

Arguments:

`resource` The computation resource.

Returns: The SshResourceClient object.

`SshResourceClient$getAllowedCommands()`: Get the command names that can be executed.

Usage:

```
SshResourceClient$getAllowedCommands()
```

Returns: A character vector

`SshResourceClient$getConnection()`: Get or create the SSH connection object, for raw interaction.

Usage:

```
SshResourceClient$getConnection()
```

Returns: The SSH connection object.

`SshResourceClient$downloadFile()`: Download one or more files (wilcard * is supported in the file name (which can be a directory))

Usage:

```
SshResourceClient$downloadFile(file, to = ".", verbose = FALSE)
```

Arguments:

`file` The file path(s) to download, either absolute or relative to the working directory.

`to` The download destination.

`verbose` If TRUE, details the file operations on the console output.

Returns: The paths of the files having been downloaded.

`SshResourceClient$uploadFile()`: Upload one or more files

Usage:

```
SshResourceClient$uploadFile(file, to = ".", verbose = FALSE)
```

Arguments:

`file` The file or vector of files to upload.

`to` The upload destination, either absolute or relative to working directory.

`verbose` If TRUE, details the file operations on the console output.

Returns: The paths of the files having been uploaded.

`SshResourceClient$tempDir()`: Get connection's temporary directory in the remote server, create it if it does not exists.

Usage:

```
SshResourceClient$tempDir()
```

Returns: The path to the temporary directory.

`SshResourceClient$removeTempDir()`: Remove the connection's temporary directory from the remote server, if it was defined.

Usage:

```
SshResourceClient$removeTempDir()
```

`SshResourceClient$exec()`: Executes a ssh command.

Usage:

```
SshResourceClient$exec(command, params = NULL, test = FALSE)
```

Arguments:

`command` The command name.

`params` A character vector of arguments to pass.

`test` If TRUE, the command is printed but not executed (for debugging).

Returns: The command execution result object.

`SshResourceClient$close()`: Close the SSH connection.

Usage:

```
SshResourceClient$close()
```

`SshResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
SshResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`SshResourceResolver` *SSH Resource resolver*

Description

The resource is a computation unit, accessible through SSH, i.e. which URL scheme is "ssh".

Format

A R6 object of class SshResourceResolver

Super class

`ResourceResolver` -> `SshResourceResolver`

Methods

Public methods:

- SshResourceResolver$isFor()
- SshResourceResolver$newClient()
- SshResourceResolver$clone()

SshResourceResolver$isFor(): Check that the provided resource is a computation resource accessible by ssh commands. The resource URL scheme is expected to be "ssh".

Usage:

```
SshResourceResolver$isFor(x)
```

Arguments:

x The resource object.

Returns: A logical.

SshResourceResolver$newClient(): Create a SshResourceClient instance from the provided resource.

Usage:

```
SshResourceResolver$newClient(x)
```

Arguments:

x A valid resource object.

Returns: A SshResourceClient object.

SshResourceResolver$clone(): The objects of this class are cloneable with this method.

Usage:

```
SshResourceResolver$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

TidyFileResourceClient

Tidy file resource client

Description

Connects to a file and use one of the tidyverse reader.

Format

A R6 object of class TidyFileResourceClient

Super classes

ResourceClient -> FileResourceClient -> TidyFileResourceClient

Methods

Public methods:

- `TidyFileResourceClient$new()`
- `TidyFileResourceClient$asDataFrame()`
- `TidyFileResourceClient$clone()`

`TidyFileResourceClient$new()`: Create a TidyFileResourceClient instance.

Usage:

```
TidyFileResourceClient$new(resource)
```

Arguments:

`resource` A valid resource object.

Returns: A TidyFileResourceClient object.

`TidyFileResourceClient$asDataFrame()`: Coerce the resource value extracted from the file in tidy format to a data.frame.

Usage:

```
TidyFileResourceClient$asDataFrame(...)
```

Arguments:

`...` Additional parameters to as.data.frame (not used yet).

Returns: A data.frame (more specifically a tibble).

`TidyFileResourceClient$clone()`: The objects of this class are cloneable with this method.

Usage:

```
TidyFileResourceClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`TidyFileResourceResolver`

Tidy file Resource resolver

Description

The resource is a file and data format is handled by a reader from tidyverse. The data format is one of: csv (comma delimiter), csv2 (semicolon delimiter), tsv (tab delimiter), ssv (space delimiter), delim (delim parameter to be specified in the URL, default is space char), spss, sav, por, stata, dta, sas, xpt, excel, xls, xlsx.

Format

A R6 object of class TidyFileResourceResolver

Methods

`$new()` Create new TidyFileResourceResolver instance. `$isFor(x)` Get a logical that indicates that the resolver is applicable to the provided resource object. `$newClient()` Make a client for the provided resource.

Super class

`ResourceResolver` -> `TidyFileResourceResolver`

Methods

Public methods:

- `TidyFileResourceResolver$isFor()`
- `TidyFileResourceResolver$newClient()`
- `TidyFileResourceResolver$clone()`

`TidyFileResourceResolver$isFor()`: Check that the provided resource has a URL that locates a tidy data file: the resource can be accessed as a file and the resource format is one of "csv", "csv2", "tsv", "delim", "ssv", "spss", "sav", "por", "stata", "dta", "sas", "xpt", "excel", "xls" or "xlsx" (case is ignored).

Usage:

```
TidyFileResourceResolver$isFor(x)
```

Arguments:

`x` The resource object to validate.

Returns: A logical.

`TidyFileResourceResolver$newClient()`: Creates a TidyFileResourceClient instance from provided resource.

Usage:

```
TidyFileResourceResolver$newClient(x)
```

Arguments:

`x` A valid resource object.

Returns: A TidyFileResourceClient object.

`TidyFileResourceResolver$clone()`: The objects of this class are cloneable with this method.

Usage:

```
TidyFileResourceResolver$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`unregisterDBIResourceConnector`
Remove a DBI resource connector from the registry

Description

Remove any DBI resource connectors with the provided class name.

Usage

```
unregisterDBIResourceConnector(x)
```

Arguments

x	The DBI resource connector class name to unregister.

`unregisterFileResourceGetter`
Remove a file resource getter from the registry

Description

Remove any file resource getters with the provided class name.

Usage

```
unregisterFileResourceGetter(x)
```

Arguments

x	The file resource getter class name to unregister.

unregisterResourceResolver *Remove a resource resolver from the registry*

Description

Remove any resolvers with the provided class name.

Usage

```
unregisterResourceResolver(x)
```

Arguments

x	The resource resolver class name to unregister.

Examples

```
## Not run:
resourcer::unregisterResourceResolver("MyFileFormatResourceResolver")

## End(Not run)
```

Index

as.data.frame.resource, 3
as.data.frame.ResourceClient, 3
as.resource.data.frame, 4
as.resource.object, 4
as.resource.tbl, 5

CommandResourceClient, 5, *36*, *42*

DBIResourceConnector, 6, *16*, *22, 23*, *39*

FileResourceClient, 8, *25*, *27*, *45*
FileResourceGetter, 9, *12, 13*, *15*, *21*, *35*
findDBIResourceConnector, 10
findFileResourceGetter, 11

getDBIResourceConnectors, 11
getFileResourceGetters, 11
getResourceResolvers, 12
GridFsFileResourceGetter, 12

HttpFileResourceGetter, 13

LocalFileResourceGetter, 15

MariaDBResourceConnector, 16

newResource, 17
newResourceClient, 18
NoSQLResourceClient, 18
NoSQLResourceResolver, 20

OpalFileResourceGetter, 21

PostgresResourceConnector, 22
PrestoResourceConnector, 23

RDataFileResourceClient, 25
RDataFileResourceResolver, 26
RDSFileResourceClient, 27
RDSFileResourceResolver, 28
registerDBIResourceConnector, 29
registerFileResourceGetter, 30
registerResourceResolver, 30
resolveResource, 31
ResourceClient, *5*, *8*, *18*, *25*, *27*, 31, *36*, *40*, *42*, *45*
ResourceResolver, *20*, *26*, *28*, 33, *37*, *41*, *44*, *47*

ScpFileResourceGetter, 35
ShellResourceClient, 36
ShellResourceResolver, 37
SparkResourceConnector, 38
SQLResourceClient, 40
SQLResourceResolver, 41
SshResourceClient, 42
SshResourceResolver, 44

TidyFileResourceClient, 45
TidyFileResourceResolver, 46

unregisterDBIResourceConnector, 48
unregisterFileResourceGetter, 48
unregisterResourceResolver, 49