Package ‘resourcer’

November 3, 2020

Type Package

Title Resource Resolver

Version 1.1.0

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)

LazyData true

Depends R (>= 3.5.0), R6, httr

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

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

RoxygenNote 7.1.1

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

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

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

Repository CRAN

Date/Publication 2020-11-03 10:00:02 UTC

R topics documented:

- 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 .................................................. 7
FileResourceGetter .................................................. 9
findDBIResourceConnector ........................................ 10
findFileResourceGetter ............................................ 11
getDBIResourceConnectors ........................................ 11
getFileResourceGetters ........................................... 11
getResourceResolvers ............................................. 12
GridFsFileResourceGetter ........................................ 12
HttpFileResourceGetter ........................................... 14
LocalFileResourceGetter ........................................... 15
MariaDBResourceConnector ....................................... 16
newResource .......................................................... 18
newResourceClient .................................................. 18
NoSQLResourceClient ............................................... 19
NoSQLResourceResolver .......................................... 21
OpalFileResourceGetter ........................................... 22
PostgresResourceConnector ....................................... 23
PrestoResourceConnector .......................................... 25
RDataFileResourceClient ......................................... 26
RDataFileResourceResolver ....................................... 27
RDSFileResourceClient ............................................ 28
RDSFileResourceResolver ......................................... 28
registerDBIResourceConnector .................................. 31
registerFileResourceGetter ...................................... 31
registerResourceResolver ........................................ 32
resolveResource .................................................... 32
ResourceClient ...................................................... 33
ResourceResolver ................................................... 35
ScpFileResourceGetter ............................................ 36
ShellResourceClient ............................................... 38
ShellResourceResolver ............................................ 39
SparkResourceConnector ......................................... 40
SQLResourceClient ............................................... 42
SQLResourceResolver ............................................. 42
SshResourceClient .................................................. 45
SshResourceResolver ............................................... 47
TidyFileResourceClient .......................................... 48
TidyFileResourceResolver ........................................ 49
unregisterDBIResourceConnector ................................. 51
unregisterFileResourceGetter ................................... 51
unregisterResourceResolver ...................................... 52

Index 53
**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**

```r
## 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**

```r
## 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.

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.

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

Value

a dplyr's tbl

CommandResourceClient

Command resource client

Description

Command resource client

Command resource client

Format

A R6 object of class CommandResourceClient

Details

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

Super class

resourcer::ResourceClient -> CommandResourceClient
Methods

Public methods:
- CommandResourceClient$new()
- CommandResourceClient$clone()

Method new(): Creates a new CommandResourceClient instance

Usage:
CommandResourceClient$new(resource)

Arguments:
resource A valid resource object.

Returns: A CommandResourceClient object.

Method 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

DBI resource connector

Format

A R6 object of class DBIResourceConnector

Details

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

Methods

Public methods:
- DBIResourceConnector$new()
- DBIResourceConnector$isFor()
- DBIResourceConnector$createDBIConnection()
- DBIResourceConnector$closeDBIConnection()
- DBIResourceConnector$clone()
Method new(): Creates a new DBIResourceConnector instance

Usage:
DBIResourceConnector$new()

Returns: A DBIResourceConnector object.

Method isFor(): Check that the provided parameter is of class "resource".

Usage:
DBIResourceConnector$isFor(resource)

Arguments:
resource The resource object to validate.

Returns: A logical.

Method 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.

Method closeDBICOnnection(): Disconnect the DBI connection.

Usage:
DBIResourceConnector$closeDBICOnnection(conn)

Arguments:
conn A DBI connection object.

Method 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
File resource client
File resource client

Format
A R6 object of class FileResourceClient
Details

Base class that connects to a file using a FileResourceGetter.

Super class

resourcer::ResourceClient -> FileResourceClient

Methods

Public methods:

• FileResourceClient$new()
• FileResourceClient$downloadFile()
• FileResourceClient$close()
• FileResourceClient$clone()

Method 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.

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

Usage:
FileResourceClient$downloadFile()

Returns: The local path to the downloaded file.

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

Usage:
FileResourceClient$close()

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

Usage:
FileResourceClient$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
Description

File resource getter

Format

A R6 object of class FileResourceGetter

Details

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.

Methods

Public methods:

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

Method `new()`:

 Creates a new FileResourceGetter instance.

Usage:
FileResource Getter$new()


Method `isFor()`:

 Check that the provided parameter is of class "resource" and has a format defined.

Usage:
FileResource Getter$ isFor(resource)

Arguments:
resource The resource object to validate.

Returns: A logical.

Method `downloadFile()`:

Stub function which subclasses will implement to make a "resource.file" object from a resource.

Usage:
findDBIResourceConnector

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**

```python
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**

```python
getDBIResourceConnectors()
```

---

**getFileResourceGetters**

*Get file resource getters registry*

**Description**

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

**Usage**

```python
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**
GridFS file resource getter

**Format**
A R6 object of class GridFsFileResourceGetter

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

**Super class**
resourcer::FileResourceGetter -> GridFsFileResourceGetter
Methods

Public methods:

• GridFsFileResourceGetter$new()
• GridFsFileResourceGetter$isFor()
• GridFsFileResourceGetter$downloadFile()
• GridFsFileResourceGetter$clone()

Method new(): Creates a new GridFsFileResourceGetter instance.

Usage:
GridFsFileResourceGetter$new()


Method 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.

Method 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.

Method 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

HTTP file resource getter

Format

A R6 object of class HttpFileResourceGetter

Details

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

Super class

resourcer::FileResourceGetter -> HttpFileResourceGetter

Methods

Public methods:

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


Usage:

HttpFileResourceGetter$new()


Method 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.

Method 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.

**Method** clone(): The objects of this class are cloneable with this method.

Usage:
HttpFileResourceGetter$clone(deep = FALSE)

Arguments:
deeplWhether to make a deep clone.

---

**LocalFileResourceGetter**

*Local file resource getter*

**Description**

Local file resource getter

**Format**

A R6 object of class LocalFileResourceGetter

**Details**

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

**Super class**

`resourcer::FileResourceGetter` -> LocalFileResourceGetter

**Methods**

**Public methods:**

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

**Method** new(): Creates a new LocalFileResourceGetter instance.

Usage:
LocalFileResourceGetter$new()


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

*Usage:*

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

*Arguments:*

- `resource` The resource object to validate.

*Returns*: A logical.

**Method** `downloadFile()`: Make a "resource.file" object from a local file resource.

*Usage:*

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

*Arguments:*

- `resource` A valid resource object.
- `...` Unused additional parameters.

*Returns*: The "resource.file" object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

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

*Arguments:*

- `deep` Whether to make a deep clone.

---

MariaDBResourceConnector

**MariaDB DBI resource connector**

**Description**

MariaDB DBI resource connector

**Format**

A R6 object of class MariaDBResourceConnector

**Details**

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

Super class

\texttt{resourcer::DBIResourceConnector -> MariaDBResourceConnector}

Methods

Public methods:

- \texttt{MariaDBResourceConnector\new()}
- \texttt{MariaDBResourceConnector\isFor()}
- \texttt{MariaDBResourceConnector\createDBICConnectionString()}
- \texttt{MariaDBResourceConnector\clone()}

Method \texttt{new()}: Creates a new MariaDBResourceConnector instance.

\textit{Usage:}

\texttt{MariaDBResourceConnector\new()}

\textit{Returns:} A MariaDBResourceConnector object.

Method \texttt{isFor()}: Check that the provided resource has a URL that locates a MySQL or MariaDB object: the URL scheme must be "mysql" or "mariadb".

\textit{Usage:}

\texttt{MariaDBResourceConnector\isFor(resource)}

\textit{Arguments:}

resource The resource object to validate.

\textit{Returns:} A logical.

Method \texttt{createDBICConnectionString()}: Creates a DBI connection object from a resource.

\textit{Usage:}

\texttt{MariaDBResourceConnector\createDBICConnectionString(resource)}

\textit{Arguments:}

resource A valid resource object.

\textit{Returns:} A DBI connection object.

Method \texttt{clone()}: The objects of this class are cloneable with this method.

\textit{Usage:}

\texttt{MariaDBResourceConnector\clone(deep = FALSE)}

\textit{Arguments:}

depth Whether to make a deep clone.
newResourceClient

Create a Resource

Description

Creates a new Resource structure.

Usage

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

Arguments

name
Optional 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

Create 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$getTableName()`
- `NoSQLResourceClient$close()`
- `NoSQLResourceClient$clone()`

**Method `new()`**: Creates a new `NoSQLResourceClient` instance.

*Usage:*

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

*Arguments:*

- `resource`: A valid resource object.

*Returns:* A `NoSQLResourceClient` object.

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

*Usage:*

```
NoSQLResourceClient$getConnection()
```

*Returns:* The nodbi connection object.

**Method `asDataFrame()`**: Makes a data.frame from the remote database table.

*Usage:*

```
NoSQLResourceClient$asDataFrame()
```

*Returns:* A tibble.

**Method `getDatabaseName()`**: Extract the database name from the resource URL.

*Usage:*

```
NoSQLResourceClient$getDatabaseName()
```

*Returns:* The database name.

**Method `getTableName()`**: Extract the database table name from the resource URL.

*Usage:*

```
NoSQLResourceClient$getTableName()
```

*Returns:* The database table name.

**Method `close()`**: Close the nodbi connection.

*Usage:*

```
NoSQLResourceClient$close()
```

**Method `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  \hspace{1cm} NoSQL Database Resource resolver

**Description**

NoSQL Database Resource resolver

**Format**

A R6 object of class NoSQLResourceResolver

**Details**

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

**Super class**

`resourcer::ResourceResolver` -> NoSQLResourceResolver

**Methods**

- **Public methods:**
  - `NoSQLResourceResolver$isFor()`
  - `NoSQLResourceResolver$newClient()`
  - `NoSQLResourceResolver$clone()`

**Method** `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.

**Method** `newClient()`: Creates a NoSQLResourceClient instance from provided resource.

  **Usage:**
  `NoSQLResourceResolver$newClient(x)`

  **Arguments:**
  - `x` A valid resource object.

  **Returns:** A NoSQLResourceClient object.
Method clone(): The objects of this class are cloneable with this method.

Usage:
NoSQLResourceResolver$clone(deep = FALSE)

Arguments:
deepl Whether to make a deep clone.
OpalFileResourceGetter$isNewFor(resource)

Arguments:
resource The resource object to validate.

Returns: A logical.

Method `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.

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

Usage:
OpalFileResourceGetter$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

---

**Description**

Postgres DBI resource connector

**Format**

A R6 object of class PostgresResourceConnector

**Details**

Makes a Postgres DBI connection from a resource description.

**Super class**

`resourcer::DBIResourceConnector` -> PostgresResourceConnector
Methods

Public methods:

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

**Method new():** Creates a new `PostgresResourceConnector` instance.

*Usage:*

```r
PostgresResourceConnector$new()
```

*Returns:* A `PostgresResourceConnector` object.

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

*Usage:*

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

*Arguments:*

| resource | The resource object to validate. |

*Returns:* A logical.

**Method createDBIConnection():** Creates a DBI connection object from a resource.

*Usage:*

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

*Arguments:*

| resource | A valid resource object. |

*Returns:* A DBI connection object.

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

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

*Arguments:*

| deep      | Whether to make a deep clone. |
PrestoResourceConnector

_Presto DBI resource connector_

Description

Presto DBI resource connector

Format

A R6 object of class PrestoResourceConnector

Details

Makes a Presto DBI connection from a resource description.

Super class

`resourcer::DBIResourceConnector` -> PrestoResourceConnector

Methods

Public methods:

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

Method `new()`:

Creates a new PrestoResourceConnector instance.

_Usage:_

```
PrestoResourceConnector$new()
```

_Returns:_ A PrestoResourceConnector object.

Method `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.

Method `createDBIConnection()`:

Creates a DBI connection object from a resource.

_Usage:_

```
PrestoResourceConnector$createDBIConnection(resource)

*Arguments:*
resource A valid resource object.

*Returns:* A DBI connection object.

**Method 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**

R data file resource client

**Format**

A R6 object of class RDSFileResourceClient

**Details**

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

**Super classes**

resourcer::ResourceClient -> resourcer::FileResourceClient -> RDataFileResourceClient

**Methods**

**Public methods:**

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

**Method new():** Creates a new RDataFileResourceClient instance.

*Usage:*
RDataFileResourceClient$new(resource)

*Arguments:*
RDataFileResourceResolver

Description

R data file Resource resolver

Format

A R6 object of class RDataFileResourceResolver

Details

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

Super class

resourcer::ResourceResolver -> RDataFileResourceResolver
Methods

Public methods:

• RDataFileResourceResolver$isFor()
• RDataFileResourceResolver$newClient()
• RDataFileResourceResolver$clone()

Method 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.

Method newClient(): Creates a RDataFileResourceClient instance from provided resource.

Usage:
RDataFileResourceResolver$newClient(x)

Arguments:
x A valid resource object.

Returns: A RDataFileResourceClient object.

Method 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

R object file resource client

Format

A R6 object of class RDSFileResourceClient
Details

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.

Super classes

resourcer::ResourceClient -> resourcer::FileResourceClient -> RDSFileResourceClient

Methods

Public methods:

• RDSFileResourceClient$new()
• RDSFileResourceClient$asDataFrame()
• RDSFileResourceClient$getValue()
• RDSFileResourceClient$clone()

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

Usage:
RDSFileResourceClient$new(resource)

Arguments:
resource A valid resource object.

Returns: A RDSFileResourceClient object.

Method $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.

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

Usage:
RDSFileResourceClient$getValue()

Returns: The resource value.

Method $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

R object file Resource resolver

Format

A R6 object of class RDSFileResourceResolver

Details

The resource is a RDS file.

Super class

resourcer::ResourceResolver -> RDSFileResourceResolver

Methods

Public methods:

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

Method 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.

Method newClient(): Creates a RDSFileResourceClient instance from provided resource.

Usage:
RDSFileResourceResolver$newClient(x)

Arguments:
x A valid resource object.

Returns: A RDSFileResourceClient object.
Method **clone()**: The objects of this class are cloneable with this method.

**Usage:**

```r
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**

```r
registerDBIResourceConnector(x)
```

**Arguments**

- `x` The DBI resource connector object to register.

---

**registerFileResourceGetter**

*Register a file resource getter*

**Description**

Maintain an list of **FileResourceGetters** that will be called when a new file resource getter is to be found.

**Usage**

```r
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**

```r
## 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

Resource client

Format

A R6 object of class ResourceClient

Details

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

Methods

Public methods:

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

Method new(): Creates a ResourceClient instance.

Usage:
ResourceClient$new(resource)

Arguments:
resource The resource object to be interpreted.

Returns: A ResourceClient object.

**Method** getResource(): Get the resource object.

*Usage:*
ResourceClient$getResource()

*Returns: The resource object.*

**Method** getConnection(): Get the implementation-specific object that connects to the resource.

*Usage:*
ResourceClient$getConnection()

*Returns: The connection object.*

**Method** 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.*

**Method** 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).*

**Method** 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.*

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

*Usage:*
ResourceClient$exec(...)
Additional parameters that will represent the command to execute.

Returns: A command execution result object.

Method close(): Silently closes the connection to the resource

Usage:
ResourceClient$close()

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

Usage:
ResourceClient$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

<table>
<thead>
<tr>
<th>ResourceResolver</th>
<th>Resource resolver</th>
</tr>
</thead>
</table>

Description

Resource resolver
Resource resolver

Format

A R6 object of class ResourceResolver

Details

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

Methods

Public methods:
- ResourceResolver$new()
- ResourceResolver$isFor()
- ResourceResolver$newClient()
- ResourceResolver$clone()

Method new(): Creates a new ResourceResolver instance.

Usage:
ResourceResolver$new()

Returns: A ResourceResolver object.

Method isFor(): Check that the provided object is of class "resource".

Usage:
ResourceResolver$isFor(x)

*Arguments:*  
x  The resource object to evaluate.

*Returns:*  A logical.

**Method** 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.

**Method** 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**  
SCP file resource getter

**Format**  
A R6 object of class ScpFileResourceGetter

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

**Super class**  
resourcer::FileResourceGetter -> ScpFileResourceGetter
Methods

Public methods:

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

Method new(): Creates a ScpFileResourceGetter instance.

Usage:
ScpFileResourceGetter$new()


Method 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.

Method 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.

Method 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

Description

Shell resource client
Shell resource client

Format

A R6 object of class ShellResourceClient

Details

Executes local system shell commands.

Super classes

resourcer::ResourceClient -> resourcer::CommandResourceClient -> ShellResourceClient

Methods

Public methods:

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

Method new(): Create a ShellResourceClient instance. This client will interact with a computation resource using shell commands.

Usage:
ShellResourceClient$new(resource)

Arguments:
resource The computation resource.

Returns: The ShellResourceClient object.

Method getAllowedCommands(): Get the command names that can be executed.

Usage:
ShellResourceClient$getAllowedCommands()

Returns: A character vector

Method copyFile(): Copy one or more files (wildcard * is supported in the file name (which can be a directory))
ShellResourceResolver

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.

Method 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.

Method 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
Shell Resource resolver
Shell Resource resolver

Format
A R6 object of class ShellResourceResolver

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

Super class
resourcer::ResourceResolver -> ShellResourceResolver
Methods

Public methods:

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

Method `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.

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

Usage:

`ShellResourceResolver$newClient(x)`

Arguments:

- `x` A valid resource object.

Returns: A `ShellResourceClient` object.

Method `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

Apache Spark DBI resource connector

Apache Spark DBI resource connector

Format

A R6 object of class SparkResourceConnector
SparkResourceConnector

Details

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

Super class

`resourcer::DBIResourceConnector` -> SparkResourceConnector

Methods

Public methods:

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

Method `new()`: Create a SparkResourceConnector instance.

Usage:

```
SparkResourceConnector$new()
```

Returns: A SparkResourceConnector object.

Method `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.

Method `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.

Method `closeDBIConnection()`: Close the DBI connection to Apache Spark.

Usage:

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

Arguments:

conn A DBI connection object.
**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

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

*Arguments:*

- `deep` Whether to make a deep clone.

---

**Description**

SQL database resource client

**Format**

A R6 object of class `SQLResourceClient`

**Details**

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

**Super class**

```
resourcer::ResourceClient -> SQLResourceClient
```

**Methods**

**Public methods:**

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

**Method** `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.
SQLResourceClient

Returns: The SQLResourceClient object.

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

Usage:
SQLResourceClient$getConnection()

Returns: The DBI connection object.

Method 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).

Method asTbl(): Get the SQL table as a dplyr's tbl.

Usage:
SQLResourceClient$asTbl()

Returns: A dplyr's tbl object.

Method getTableName(): Get the SQL table name from the resource URL.

Usage:
SQLResourceClient$getTableName()

Returns: The SQL table name.

Method close(): Silently close the DBI connection.

Usage:
SQLResourceClient$close()

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

Usage:
SQLResourceClient$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.
SQLResourceResolver   SQL Database Resource resolver

Description

SQL Database Resource resolver
SQL Database Resource resolver

Format

A R6 object of class SQLResourceResolver

Details

The resource is SQL database.

Super class

resourcer::ResourceResolver -> SQLResourceResolver

Methods

Public methods:

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

Method isFor(): Check that the provided resource has a registered DBIResourceConnector.

Usage:
SQLResourceResolver$isFor(x)

Arguments:

- x  The resource object to evaluate.

Returns: A logical.

Method newClient(): Creates a SQLResourceClient instance from provided resource.

Usage:
SQLResourceResolver$newClient(x)

Arguments:

- x  A valid resource object.

Returns: A SQLResourceClient object.

Method 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  

**Description**

SSH resource client

**Format**

A R6 object of class SshResourceClient

**Details**

Connects to a SSH server.

**Super classes**

resourcer::ResourceClient -> resourcer::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()

**Method** `new()`: Create a SshResourceClient instance. This client will interact with a computation resource using ssh commands.

*Usage:

SshResourceClient$new(resource)*

**Arguments:**

resource  The computation resource.

**Returns:**  The SshResourceClient object.

**Method** `getAllowedCommands()`: Get the command names that can be executed.

*Usage:
SshResourceClient

SshResourceClient$getAllowedCommands()

*Returns:* A character vector

**Method getConnection():** Get or create the SSH connection object, for raw interaction.

*Usage:*

SshResourceClient$getConnection()

*Returns:* The SSH connection object.

**Method downloadFile():** Download one or more files (wildcard * 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.

**Method 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.

**Method 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.

**Method removeTempDir():** Remove the connection’s temporary directory from the remote server, if it was defined.

*Usage:*

SshResourceClient$removeTempDir()

**Method exec():** Executes a ssh command.

*Usage:*

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

*Arguments:*

command The command name.
SshResourceResolver

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.

Method close(): Close the SSH connection.
Usage:
SshResourceClient$close()

Method 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
SSH Resource resolver

Format
A R6 object of class SshResourceResolver

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

Super class
resourcer::ResourceResolver -> SshResourceResolver

Methods
Public methods:
• SshResourceResolver$isFor()
• SshResourceResolver$newClient()
• SshResourceResolver$clone()

Method 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.

**Method newClient():** Create a SshResourceClient instance from the provided resource.

**Usage:**

SshResourceResolver$newClient(x)

**Arguments:**

x A valid resource object.

**Returns:** A SshResourceClient object.

**Method clone():** The objects of this class are cloneable with this method.

**Usage:**

SshResourceResolver$clone(deep = FALSE)

**Arguments:**

depth Whether to make a deep clone.

---

TidyFileResourceClient

*Tidy file resource client*

---

**Description**

Tidy file resource client

Tidy file resource client

**Format**

A R6 object of class TidyFileResourceClient

**Details**

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

**Super classes**

resourcer::ResourceClient -> resourcer::FileResourceClient -> TidyFileResourceClient
Methods

Public methods:

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

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

Usage:

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

Arguments:

- `resource` A valid resource object.

Returns: A TidyFileResourceClient object.

Method `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).

Method `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

Tidy file Resource resolver

Tidy file Resource resolver

Format

A R6 object of class TidyFileResourceResolver
Details

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.

Methods

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

Super class

\texttt{resourcer::ResourceResolver \rightarrow TidyFileResourceResolver}

Methods

Public methods:

- \texttt{TidyFileResourceResolver$\texttt{isFor}()}
- \texttt{TidyFileResourceResolver$\texttt{newClient}()}
- \texttt{TidyFileResourceResolver$\texttt{clone}()}

Method \texttt{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:
\texttt{TidyFileResourceResolver$\texttt{isFor}(x)}

Arguments:
- \texttt{x} The resource object to validate.

Returns: A logical.

Method \texttt{newClient():} Creates a TidyFileResourceClient instance from provided resource.

Usage:
\texttt{TidyFileResourceResolver$\texttt{newClient}(x)}

Arguments:
- \texttt{x} A valid resource object.

Returns: A TidyFileResourceClient object.

Method \texttt{clone():} The objects of this class are cloneable with this method.

Usage:
\texttt{TidyFileResourceResolver$\texttt{clone}(deep = FALSE)}

Arguments:
- \texttt{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
DBIResourceConnector, 6
FileResourceClient, 7
FileResourceGetter, 9
findDBIResourceConnector, 10
findFileResourceGetter, 11
getDBIResourceConnectors, 11
getFileResourceGetters, 11
getResourceResolvers, 12
GridFsFileResourceGetter, 12
HttpFileResourceGetter, 14
LocalFileResourceGetter, 15
MariaDBResourceConnector, 16
newResource, 18
newResourceClient, 18
NoSQLResourceClient, 19
NoSQLResourceResolver, 21
OpalFileResourceGetter, 22
PostgresResourceConnector, 23
PrestoResourceConnector, 25
RDataFileResourceClient, 26
RDataFileResourceResolver, 27
RDSFileResourceClient, 28
RDSFileResourceResolver, 30
registerDBIResourceConnector, 31
registerFileResourceGetter, 31
registerResourceResolver, 32
resolveResource, 32
ResourceClient, 33
resourcer::CommandResourceClient, 38, 45
resourcer::DBIResourceConnector, 17, 23, 25, 41
resourcer::FileResourceClient, 26, 29, 48
resourcer::FileResourceGetter, 12, 14, 15, 22, 36
resourcer::ResourceClient, 5, 8, 19, 26, 29, 38, 42, 45, 48
resourcer::ResourceResolver, 21, 27, 30, 39, 44, 47, 50
ResourceResolver, 35
ScpFileResourceGetter, 36
ShellResourceClient, 38
ShellResourceResolver, 39
SparkResourceConnector, 40
SQLResourceClient, 42
SQLResourceResolver, 44
SshResourceClient, 45
SshResourceResolver, 47
TidyFileResourceClient, 48
TidyFileResourceResolver, 49
unregisterDBIResourceConnector, 51
unregisterFileResourceGetter, 51
unregisterResourceResolver, 52