

# Package ‘s3.resourcer’

January 18, 2025

**Type** Package

**Title** S3 Resource Resolver

**Version** 1.1.2

**Description** A S3 resource is provided by Amazon Web Services S3 or a S3-compatible object store (such as Minio).

The resource can be a tidy file to be downloaded from the object store, or a data lake (such as Delta Lake) Parquet file to be read by Apache Spark.

**License** LGPL (>= 2.1)

**Depends** R6, httr, resourcer (>= 1.2), aws.s3, sparklyr

**Suggests** testthat, knitr

**BugReports** <https://github.com/obiba/s3.resourcer/issues>

**RoxygenNote** 7.2.3

**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** 2025-01-18 18:10:02 UTC

## Contents

S3FileResourceGetter . . . . .	2
S3SparkResourceConnector . . . . .	3
<b>Index</b>	<b>6</b>

---

S3FileResourceGetter *AWS S3 file resource getter*

---

### Description

AWS S3 file resource getter

AWS S3 file resource getter

### Format

A R6 object of class S3FileResourceGetter

### Details

Access a file that is in the Amazon Web Services S3 file store or in a HTTP S3 compatible file store such as Minio. For AWS S3 the host name is the bucket name. For Minio, the url will include http or https base protocol. Credentials may apply.

### Super class

[resourcer::FileResourceGetter](#) -> S3FileResourceGetter

### Methods

#### Public methods:

- [S3FileResourceGetter\\$new\(\)](#)
- [S3FileResourceGetter\\$isFor\(\)](#)
- [S3FileResourceGetter\\$downloadFile\(\)](#)
- [S3FileResourceGetter\\$clone\(\)](#)

**Method** `new()`: Creates a new S3FileResourceGetter instance.

*Usage:*

```
S3FileResourceGetter$new()
```

*Returns:* A S3FileResourceGetter object.

**Method** `isFor()`: Check that the provided resource has a URL that locates a file accessible through "s3" protocol or "s3+http" or "s3+https" protocol (i.e. using Minio implementation of the AWS S3 file store API over HTTP).

*Usage:*

```
S3FileResourceGetter$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 authentication if credentials are provided in the resource.

*Usage:*

```
S3FileResourceGetter$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:*

```
S3FileResourceGetter$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

---

S3SparkResourceConnector

*Apache Spark DBI resource connector for S3*

---

## Description

Apache Spark DBI resource connector for S3

Apache Spark DBI resource connector for S3

## Format

A R6 object of class `SparkResourceConnector`

## Details

Makes a Apache Spark connection object, that is also a DBI connection object, from a S3 resource description.

## Super classes

`resourcer::DBIResourceConnector` -> `resourcer::SparkResourceConnector` -> `S3SparkResourceConnector`

## Methods

### Public methods:

- `S3SparkResourceConnector$new()`
- `S3SparkResourceConnector$isFor()`
- `S3SparkResourceConnector$createDBIConnection()`
- `S3SparkResourceConnector$getTableNames()`

- [S3SparkResourceConnector\\$readDBTable\(\)](#)
- [S3SparkResourceConnector\\$readDBTibble\(\)](#)
- [S3SparkResourceConnector\\$closeDBIConnection\(\)](#)
- [S3SparkResourceConnector\\$clone\(\)](#)

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

*Usage:*

```
S3SparkResourceConnector$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 "s3+spark", "s3+spark+http" or "s3+spark+https".

*Usage:*

```
S3SparkResourceConnector$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:*

```
S3SparkResourceConnector$createdBIConnection(resource)
```

*Arguments:*

resource A valid resource object.

*Returns:* A DBI connection object.

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

*Usage:*

```
S3SparkResourceConnector$getTableName(resource)
```

*Arguments:*

resource A valid resource object.

*Returns:* The SQL table name.

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

*Usage:*

```
S3SparkResourceConnector$readDBTable(conn, resource)
```

*Arguments:*

conn A DBI connection object.

resource A valid resource object.

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

*Usage:*

S3SparkResourceConnector\$readDBTibble(conn, resource)

*Arguments:*

conn A DBI connection object.

resource A valid resource object.

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

*Usage:*

S3SparkResourceConnector\$closeDBIConnection(conn)

*Arguments:*

conn A DBI connection object.

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

*Usage:*

S3SparkResourceConnector\$clone(deep = FALSE)

*Arguments:*

deep Whether to make a deep clone.

# Index

`resource::DBIResourceConnector`, [3](#)  
`resource::FileResourceGetter`, [2](#)  
`resource::SparkResourceConnector`, [3](#)  
  
`S3FileResourceGetter`, [2](#)  
`S3SparkResourceConnector`, [3](#)