Package 'OhdsiSharing'

June 18, 2020

Type Package

Title Package for sharing of the results of the OHDSI tools

Version 0.2.2

Date 2020-06-18

Maintainer Martijn Schuemie <schuemie@ohdsi.org>

Description Package for sharing of the results of the OHDSI tools, with functions for encrypting results and sending results through SFTP to a central site.

URL https://ohdsi.github.io/OhdsiSharing, https://github.com/OHDSI/OhdsiSharing

BugReports https://github.com/OHDSI/OhdsiSharing/issues

Imports rJava, ParallelLogger

Suggests testthat

License Apache License

RoxygenNote 7.1.0

R topics documented:

`compressAndEncryptFolder` *Compress and encrypt a folder*

Description

Compress and encrypt a folder

Usage

```
compressAndEncryptFolder(sourceFolder, targetFileName, publicKeyFileName)
```

Arguments

`sourceFolder`	Name of the folder that must be encrypted.
`targetFileName`	Name of the file that will hold the encrypted data.
`publicKeyFileName`	Name of the file where the public key is stored.

Details

Compresses all files in a folder and its subfolders, and encrypts using the provided public key.

Examples

```
## Not run:
generateKeyPair("public.key", "private.key")

# Create a folder with some data
dir.create("test")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "test/data1.rds")
saveRDS(data, "test/data2.rds")

compressAndEncryptFolder("test", "data.zip.enc", "public.key")
decryptAndDecompressFolder("data.zip.enc", "test2", "private.key")

## End(Not run)
```

compressFolder *Compress a folder*

Description

Compress a folder

Usage

```
compressFolder(sourceFolder, targetFileName)
```

Arguments

sourceFolder	Name of the folder that must be compressed.
targetFileName	Name of the file that will hold the compressed data.

Details

Compresses all files in a folder and its subfolders, and stores it in a single zip file.

Examples

```
## Not run:
# Create a folder with some data
dir.create("test")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "test/data1.rds")
saveRDS(data, "test/data2.rds")

compressFolder("test", "data.zip")
decompressFolder("data.zip", "test2")

## End(Not run)
```

decompressFolder *Decompress a folder*

Description

Decompress a folder

Usage

```
decompressFolder(sourceFileName, targetFolder)
```

Arguments

sourceFileName	Name of the file that must be decompressed.
targetFolder	Name of the folder that will hold the extracted data.

Details

Extracts all compressed files to a folder.

Examples

```
## Not run:
# Create a folder with some data
dir.create("test")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "test/data1.rds")
saveRDS(data, "test/data2.rds")

compressFolder("test", "data.zip")
decompressFolder("data.zip", "test2")

## End(Not run)
```

`decryptAndDecompressFolder` *Decrypt and decompress a folder*

Description

Decrypt and decompress a folder

Usage

```
decryptAndDecompressFolder(sourceFileName, targetFolder, privateKeyFileName)
```

Arguments

`sourceFileName`	Name of the file that must be decrypted.
`targetFolder`	Name of the folder that will hold the unencrypted data.
`privateKeyFileName`	Name of the file where the private key is stored.

Details

Decrypts the data using the provided private key and extracts all files to a folder.

Examples

```
## Not run:
generateKeyPair("public.key", "private.key")

# Create a folder with some data
dir.create("test")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "test/data1.rds")
saveRDS(data, "test/data2.rds")
```

```
compressAndEncryptFolder("test", "data.zip.enc", "public.key")
decryptAndDecompressFolder("data.zip.enc", "test2", "private.key")

## End(Not run)
```

`decryptFile` *Decrypt a data file*

Description

Decrypt a data file

Usage

```
decryptFile(sourceFileName, targetFileName, privateKeyFileName)
```

Arguments

`sourceFileName`	Name of the file that must be decrypted.
`targetFileName`	Name of the file that will hold the unencrypted data.
`privateKeyFileName`	Name of the file where the private key is stored.

Details

Decrypts the data using the provided private key.

Examples

```
## Not run:
generateKeyPair("public.key", "private.key")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "data.rds")
encryptFile("data.rds", "data.rds.enc", "public.key")
decryptFile("data.rds.enc", "data2.rds", "private.key")

## End(Not run)
```

`encryptFile` *Encrypt a data file*

Description

Encrypt a data file

Usage

```
encryptFile(sourceFileName, targetFileName, publicKeyFileName)
```

Arguments

sourceFileName	Name of the file that must be encrypted.
targetFileName	Name of the file that will hold the encrypted data.
publicKeyFileName	Name of the file where the public key is stored.

Details

Encrypts the data using the provided public key.

Examples

```
## Not run:
generateKeyPair("public.key", "private.key")
data <- data.frame(x = runif(1000), y = 1:1000)
saveRDS(data, "data.rds")
encryptFile("data.rds", "data.rds.enc", "public.key")

## End(Not run)
```

---

generateKeyPair *Create a public-private key pair*

Description

Create a public-private key pair

Usage

```
generateKeyPair(publicKeyFileName, privateKeyFileName)
```

Arguments

publicKeyFileName	Name of the file where the public key should be stored.
privateKeyFileName	Name of the file where the private key should be stored.

Details

Creates an RSA 4096-bit public-private key pair. The public key can be used to encrypt data, and only with the private key can the data be decrypted.

Examples

```
## Not run:
generateKeyPair("public.key", "private.key")

## End(Not run)
```

sftpCd *Change the current working director*

Description

Change the current working director

Usage

```
sftpCd(sftpConnection, remoteFolder)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFolder	The folder on the server to change to.

sftpConnect *Connect to the OHDSI SFTP server*

Description

Connect to the OHDSI SFTP server

Usage

```
sftpConnect(privateKeyFileName, userName)
```

Arguments

privateKeyFileName	A character string denoting the path to an RSA private key.
userName	A character string containing the user name.

Value

An SftpConnection object

sftpDisconnect *Disconnect from the OHDSI SFTP server.*

Description

Disconnect from the OHDSI SFTP server.

Usage

```
sftpDisconnect(sftpConnection)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.

sftpGetFiles *Get one or more files from the SFTP server*

Description

Get one or more files from the SFTP server

Usage

```
sftpGetFiles(
  sftpConnection,
  remoteFileNames,
  localFolder = getwd(),
  localFileNames = file.path(localFolder, remoteFileNames)
)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFileNames	The name of the file(s) to get from the server.
localFolder	The path of a local folder where all files will be stored. Is ignored if localFileNames is provided.
localFileNames	The name the file(s) should have locally. If not provided, the files will be given the same names as on the server.

sftpLs *List the files in folder on the server.*

Description

List the files in folder on the server.

Usage

```
sftpLs(sftpConnection, remoteFolder = "./")
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFolder	The folder on the server. Defaults to the current folder.

Value

A data frame with two columns: the file names, and the file types (directory, link, or file).

sftpMkdir *Make a directory*

Description

Make a directory

Usage

```
sftpMkdir(sftpConnection, remoteFolder)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFolder	The folder on the server to create.

sftpPutFile *Put a file on the SFTP server*

Description

Put a file on the SFTP server

Usage

```
sftpPutFile(
  sftpConnection,
  localFileName,
  remoteFileName = basename(localFileName)
)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
localFileName	The path to the local file to upload.
remoteFileName	The name the file should have on the server.

sftpRename *Rename a file or folder*

Description

Rename a file or folder

Usage

```
sftpRename(sftpConnection, oldRemoteFilename, newRemoteFilename)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
oldRemoteFilename	The file on the server to rename.
newRemoteFilename	The new file name.

sftpRm *Remove one or more files*

Description

Remove one or more files

Usage

```
sftpRm(sftpConnection, remoteFiles)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFiles	The file(s) on the server to remove.

sftpRmdir *Remove a directory*

Description

Remove a directory

Usage

```
sftpRmdir(sftpConnection, remoteFolder)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.
remoteFolder	The folder on the server to remove.

sftpUploadFile *Upload a single file to the OHDSI SFTP server*

Description

This function combines calls to the sftpConnect, sftpPutFile, and sftpDisconnect functions. A random string will be prefixed to the file name to prevent overwriting existing files on the server.

Usage

```
sftpUploadFile(privateKeyFileName, userName, remoteFolder = ".", fileName)
```

Arguments

privateKeyFileName	A character string denoting the path to an RSA private key.
userName	A character string containing the user name.
remoteFolder	The remote folder to upload the file to.
fileName	A character string denoting the path to file to upload.

sftPwd *Get the present working directory*

Description

Get the present working directory

Usage

```
sftPwd(sftpConnection)
```

Arguments

sftpConnection	An SftpConnection object as created by the sftpConnect function.

Value

A character string representing the current remote folder name.

Index