Package: rdav (via r-universe)

October 26, 2026

Type Package

Title Simple 'WebDAV' Client

Version 0.3.0

Date 2025-12-02

Maintainer Gunther Krauss <guntherkrauss@uni-bonn.de>

Description A simple 'WebDAV' client that provides functions to fetch and send files or folders to servers using the 'WebDAV' protocol (see 'RFC' 4918 <https://www.rfc-editor.org/rfc/rfc4918>). Only a subset of the protocol is implemented (e.g. file locks are not yet supported).

License GPL-2

URL https://github.com/gk-crop/rdav

BugReports https://github.com/gk-crop/rdav/issues

Depends R (>= 4.1.0)

Imports httr2 (>= 1.2.0), xml2

Suggests utils, keyring, knitr, rmarkdown, testthat (>= 3.0.0)

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs libxml2-dev libssl-dev

Repository https://gk-crop.r-universe.dev

Date/Publication 2025-12-03 15:22:59 UTC

RemoteUrl https://github.com/gk-crop/rdav

RemoteRef HEAD

RemoteSha 9d04c9807654f17f34d7c41c3894e3f46fe9b36a

Contents

ncl_baseurl . . . 2
ncl_shareurl . . . 3
ncl_shareurl_from_publicurl . . . 4
ncl_username_from_url . . . 4
ocs_create_share . . . 5
ocs_delete_share . . . 8
ocs_find_users . . . 8
ocs_modify_share . . . 9
ocs_send_mail . . . 10
ocs_share_info . . . 11
ocs_shares_extended . . . 11
rdav . . . 13
wd_connect . . . 14
wd_copy . . . 15
wd_delete . . . 16
wd_dir . . . 16
wd_download . . . 17
wd_getwd . . . 18
wd_isdir . . . 18
wd_mkdir . . . 19
wd_move . . . 20
wd_setwd . . . 20
wd_upload . . . 21

Index **23**

`ncl_baseurl` *Builds the Nextcloud base URL for a nextcloud server from host and user names*

Description

Builds the Nextcloud base URL for a nextcloud server from host and user names

Usage

```
ncl_baseurl(hostname, username, path_prefix = "")
```

Arguments

`hostname`	host name
`username`	user name
`path_prefix`	optional path to webdav directory

Value

base url to use with wd_connect

Examples

```
ncl_baseurl("example.com","johndoe")
ncl_baseurl("example.com","johndoe","sub/dir/")
```

`ncl_shareurl` *Builds the Nextcloud base URL for a public share from host and user names*

Description

Builds the Nextcloud base URL for a public share from host and user names

Usage

```
ncl_shareurl(hostname, username, path_prefix = "")
```

Arguments

`hostname`	host name
`username`	user name
`path_prefix`	ptional path to webdav directory

Value

share url to use with wd_connect

Examples

```
ncl_shareurl("example.com", "johndoe")
```

ncl_shareurl_from_publicurl *Creates the Nextcloud base URL for a share from a public share url*

Description

Creates the Nextcloud base URL for a share from a public share url

Usage

```
ncl_shareurl_from_publicurl(url)
```

Arguments

url	link to public share

Value

share url to use with wd_connect

Examples

```
ncl_shareurl_from_publicurl("https://example.com/s/87d7edad/")
```

ncl_username_from_url *Extracts the user name from a Nextcloud Base URL*

Description

Extracts the user name from a Nextcloud Base URL

Usage

```
ncl_username_from_url(url)
```

Arguments

url	base or share url

Value

user name

Examples

```
ncl_username_from_url("https://example.com/remote.php/dav/files/johndoe")
```

ocs_create_share *Creates a share*

Description

Creates different share types:

Usage

```
ocs_create_share(
  req,
  path,
  share_type,
  share_with = NULL,
  password = NULL,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  label = "",
  note = "",
  send_mail = FALSE,
  attributes = NULL
)

ocs_create_share_link(
  req,
  path,
  password = NULL,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  note = "",
  label = ""
)

ocs_create_share_mail(
  req,
  path,
  email,
  password = NULL,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  note = "",
  label = "",
  send_mail = TRUE
)
```

```
ocs_create_share_user(
  req,
  path,
  user,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  note = "",
  label = "",
  send_mail = TRUE
)

ocs_create_share_group(
  req,
  path,
  group,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  note = "",
  label = "",
  send_mail = TRUE
)

ocs_create_share_federated(
  req,
  path,
  cloud_id,
  permissions = 1,
  public_upload = FALSE,
  expire_date = NULL,
  note = "",
  label = "",
  send_mail = TRUE
)
```

Arguments

req	WebDAV request as returned by wd_connect
path	folder or file path
share_type	integer 0:user, 1:group, 3:link, 4:e-mail, 6:federated
share_with	depending on share type: user id, group id, e-mail address or federated cloud id (only for ocs_create_share)
password	optional password for link and e-mail shares
permissions	integer or char vector (1:read, 2:update, 4:create, 8:delete, 16:share), default is 1:read

public_upload	TRUE if public upload should be enabled
expire_date	expiration date as string in the format YYYY-MM-DD
label	label for the share
note	note for the share
send_mail	if TRUE the user is notified via e-mail
attributes	optional attributes
email	e-mail address (only for ocs_create_share_mail)
user	id of the user (only for ocs_create_share_user)
group	id of the group (only for ocs_create_share_group)
cloud_id	cloud id (only for ocs_create_share_federated)

Details

- ocs_create_share - generic method that takes share type as argument
- ocs_create_share_user - creates a share for a nextcloud user
- ocs_create_share_group - creates a share for a nextcloud group
- ocs_create_share_link - create a public share link
- ocs_create_share_mail - creates an e-mail share
- ocs_create_share_federated - creates a federated share

Notice: if protecting a public link or e-mail share with a password, make sure that the password meets the services' password policy.

Share permissions can be given as vector of integers or characters, e.g. c(1, 2, 8) or c("read", "update", "delete"), or as their sum or concatenation, e.g. 11 or "read,update,delete".

Value

information for the newly created share as data.frame

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_create_share(r, "myfolder/share", 4)
ocs_create_share_link(r, "myfolder/share")
ocs_create_share_mail(r, "myfolder/share", "jack@example.com")
ocs_create_share_user(r, "myfolder/share", "jackdoe", c("read", "update"))


## End(Not run)
```

ocs_delete_share *Deletes a share*

Description

Deletes a share

Usage

```
ocs_delete_share(req, id)
```

Arguments

req	WebDAV request as returned by wd_connect
id	share id

Value

invisible boolean TRUE on success and FALSE on failure

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_delete_share(r, 12342)

## End(Not run)
```

ocs_find_users *Search for users one could share data*

Description

Search for users one could share data

Usage

```
ocs_find_users(req, search, lookup = FALSE, as_df = TRUE)
```

Arguments

req	WebDAV request as returned by wd_connect
search	search string (e.g. user name, e-mail, group name)
lookup	if TRUE (and supported) search on Nextcloud lookup server
as_df	if TRUE (default) a data.frame is returned, else a list of IDs

Value

data.frame with user informations or named vector of user ids.

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_find_users(r, "Doe, Jack")

## End(Not run)
```

`ocs_modify_share` *Modifies properties of a share*

Description

If a parameter omitted or is `NULL`, then the coresponding property is not modified.

Usage

```
ocs_modify_share(
  req,
  id,
  password = NULL,
  permissions = NULL,
  public_upload = NULL,
  expire_date = NULL,
  label = NULL,
  note = NULL,
  send_mail = NULL,
  attributes = NULL
)
```

Arguments

`req`	WebDAV request as returned by `wd_connect`
`id`	share id
`password`	optional password for link and e-mail shares
`permissions`	integer or char vector (1:read, 2:update, 4:create, 8:delete, 16:share), default is 1:read
`public_upload`	TRUE if public upload should be enabled
`expire_date`	expiration date as string in the format YYYY-MM-DD
`label`	label for the share
`note`	note for the share
`send_mail`	if TRUE the user is notified via e-mail
`attributes`	optional attributes

Details

Share permissions can be given as vector of integers or characters, e.g. `c(1, 2, 8)` or `c("read", "update", "delete")`, or as their sum or concatenation, e.g. 11 or `"read,update,delete"`.

Value

data.frame with the share properties

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_modify_share(r, 12345, permissions = 31, expire_date = "2025-11-01")

## End(Not run)
```

---

ocs_send_mail *Notifies the user of a mail share*

Description

Notifies the user of a mail share

Usage

```
ocs_send_mail(req, id, password = NULL)
```

Arguments

`req`	WebDAV request as returned by `wd_connect`
`id`	share id
`password`	password of the share if it is password protected

Value

invisible TRUE on success or FALSE on failure

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")

# add a password to a mail share and notify the user
ocs_modify_share(r, 12342, password = "super_secret")
ocs_send_mail(r, 12342, password = "super_secret")

## End(Not run)
```

ocs_share_info *Get infor for a specific share*

Description

Get infor for a specific share

Usage

```
ocs_share_info(req, id, columns = NULL)
```

Arguments

req	WebDAV request as returned by wd_connect
id	share id
columns	column names that should be included into the result (default NULL includes all)

Value

one row data.frame with share properties

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_share_info(r, 158742)

## End(Not run)
```

ocs_shares_extended *Returns information for shares*

Description

ocs_shares_extended returns extended information for shares. ocs_shares returns the shares of a file or folder, ocs_child_shares the shares of the files and subfolders of the given path.

Usage

```
ocs_shares_extended(
  req,
  path = "",
  as_df = TRUE,
  columns = NULL,
  subfiles = TRUE,
  reshares = FALSE
```

```
)

ocs_child_shares(
  req,
  path = "",
  as_df = TRUE,
  columns = c("share_type", "item_type", "permissions", "label", "uid_owner",
    "share_with_displayname")
)

ocs_shares(
  req,
  path = "/",
  as_df = TRUE,
  columns = c("share_type", "item_type", "permissions", "label", "uid_owner",
    "share_with_displayname")
)
```

Arguments

req	WebDAV request as returned by wd_connect
path	folder or file path
as_df	if TRUE (default) a data.frame is returned, else a list of IDs
columns	column names that should be included into the result (default NULL includes all)
subfiles	list shares of subfolders
reshares	include shares from others

Value

data.frame or named vector of IDs

Examples

```
## Not run:
r <- wd_connect("https://example.com/remote.php/dav/files/johndoe")
ocs_shares_extended(r, "myfolder/shares")
ocs_shares(r, "myfolder/shares")
ocs_child_shares(r, "myfolder")

## End(Not run)
```

rdav *rdav: Interchange Files With 'WebDAV' Servers*

Description

Provides functions to interchange files with WebDAV servers

Details

- download a file or a directory (recursively) from a WebDAV server
- upload a file or a directory (recursively) to a WebDAV server
- copy, move, delete files or directories on a WebDAV server
- list directories on the WebDAV server

Notice: when uploading or downloading files, they are overwritten without any warnings.

Provides additional functions for Nextcloud servers for managing shares.

Author(s)

Gunther Krauss

See Also

Useful links:

- https://github.com/gk-crop/rdav
- Report bugs at https://github.com/gk-crop/rdav/issues

Examples

```
## Not run:
# establish a connection, you will be asked for a password
r <- wd_connect("https://example.com/remote.php/dav/files/user","user")

# show files / directoriess in main directory
wd_dir(r)

# lists 'subdir', returns a dataframe
wd_dir(r, "subdir", as_df = TRUE)

# create directory 'mydirectory' on the server
wd_mkdir(r,"mydirectory")

# upload the local file testfile.R to the subdirectory 'mydirectory'
wd_upload(r, "testfile.R", "mydirectory/testfile.R")

# download content of 'mydirectory' from the server and
# store it in 'd:/data/fromserver' on your computer
```

```
wd_download(r, "mydirectory", "d:/data/fromserver")


## End(Not run)
```

---

wd_connect *Establishes a connection to a WebDAV server*

Description

Creates and authenticate a request handle to the WebDAV server

Usage

```
wd_connect(
  url,
  username = ncl_username_from_url(url),
  password = NULL,
  directory = "/"
)
```

Arguments

url	url of the WebDAV directory
username	username - if not given, it will be derived from the url
password	password - if not given, you will be asked for it
directory	path to use as working directory

Details

Notice: it's not recommended to write the password as plain text. Either omit the parameter (then you will be asked to enter a password interactively) or use for example the system credential store via keyring package.

Value

a httr2 request to the WebDAV server location

Examples

```
## Not run:
# establish a connection, you will be asked for a password

r <- wd_connect("https://example.com/remote.php/dav/files/myname","myname")


# establish a connection, use keyring package to retrieve the password
```

```
keyring::key_set("dav", "myname") # call only once

r <- wd_connect("https://example.com/remote.php/dav/files/myname",
                "myname",
                keyring::key_get("dav", "myname"))

## End(Not run)
```

---

wd_copy *Copies a file or directory on the WebDAV server*

Description

Copies a file or directory on the WebDAV server

Usage

```
wd_copy(req, source, target, overwrite = TRUE)
```

Arguments

req	request handle obtained from wd_connect
source	path of the source on the server
target	path of the target on the server
overwrite	overwrites files when TRUE (default)

Value

TRUE on success, FALSE on failure (invisibly)

Examples

```
## Not run:

wd_copy(r, "testfile.R", "testfile_old.R")


## End(Not run)
```

wd_delete — *Deletes a file or directory (collection) on WebDAV server*

Description

Deletes a file or directory (collection) on WebDAV server

Usage

```
wd_delete(req, file)
```

Arguments

req	request handle obtained from wd_connect
file	path to file or directory to delete on the server

Value

TRUE on success, FALSE on failure (invisibly)

Examples

```
## Not run:

wd_delete(r, "testfile.R")


## End(Not run)
```

wd_dir — *Lists the content of a WebDAV directory*

Description

Lists the content of a WebDAV directory

Usage

```
wd_dir(req, directory = "", full_names = FALSE, as_df = FALSE)
```

Arguments

req	request handle obtained from wd_connect
directory	directory path
full_names	if TRUE, the directory path is prepended to the file names to give a relative file path (relevant only if as_df is FALSE)
as_df	if TRUE outputs a data.frame with file information

Value

a vector of filenames or a dataframe (when as_df is TRUE) with detailed file information (filename, path, isdir, size, lastmodified)

Examples

```
## Not run:

# lists names of files and directories in the main directory
wd_dir(r)

# lists names of files and directories in the subdirectory "mydirectory"
wd_dir(r, "mydirectory")

# lists names of files and directories with the relative path
wd_dir(r, "mydirectory", full_names=TRUE)

# returns a data.frame with the columns filename, size and isdir (whether
# it's a directory or file
wd_dir(r, "mydirectory", as_df=TRUE)


## End(Not run)
```

`wd_download` *Fetches a file or directory (recursively) from the WebDAV server*

Description

Directories are downloaded recursively. If the source is a file and the target a directory, then the file is downloaded to the target directory. If the target is omitted, then the file or directory name (`basename`) will be used.

Usage

```
wd_download(req, source, target = "")
```

Arguments

`req`	request handle obtained from `wd_connect`
`source`	path to source file or directory on server
`target`	path to local target file or directory, if omitted the file or directory name will be used, if source is a file and target a directory then the file will be put into the target directory

Value

vector of downloaded files (invisibly)

Examples

```
## Not run:

wd_download(r, "weatherfiles", "d:/data/weather")
wd_download(r, "test/xyz.txt", "d:/data/abc.txt")


## End(Not run)
```

---

wd_getwd *Get the current WebDAV working directory*

Description

Get the current WebDAV working directory

Usage

```
wd_getwd(req)
```

Arguments

req	request handle obtained from wd_connect

Value

name of the WebDAV directory

Examples

```
## Not run:
wd_getwd(req)

## End(Not run)
```

---

wd_isdir *Checks if the resource on WebDAV is a directory*

Description

Checks if the resource on WebDAV is a directory

Usage

```
wd_isdir(req, directory, silent = FALSE)
```

Arguments

req	request handle obtained from wd_connect
directory	path to directory
silent	if FALSE a warning is given if the directory does not exists

Value

TRUE if it is a directory, FALSE else

Examples

```
## Not run:

wd_isdir(r, "testfile.R") # FALSE
wd_isdir(r, "mydirectory")   # TRUE


## End(Not run)
```

---

wd_mkdir *Creates a directory (collection) on WebDAV server*

Description

When creating a subdirectoy, all parent directories have to exist on the server.

Usage

```
wd_mkdir(req, directory)
```

Arguments

req	request handle obtained from wd_connect
directory	directory path on server

Value

TRUE on success, FALSE on failure (invisibly)

Examples

```
## Not run:

# creates 'newdir' inside the subdirectory 'existing/directory'
wd_mkdir(r, "existing/directory/newdir")


## End(Not run)
```

wd_move *Moves a file or directory on the server*

Description

Moves a file or directory on the server

Usage

```
wd_move(req, source, target, overwrite = TRUE)
```

Arguments

req	request handle obtained from wd_connect
source	path of the source on the server
target	path of the target on the server
overwrite	overwrites files when TRUE (default)

Value

TRUE on success, FALSE on failure (invisibly)

Examples

```
## Not run:

wd_move(r, "testfile.R", "testfile_old.R")


## End(Not run)
```

wd_setwd *Set working directory*

Description

If the directory path starts with a forward slash, then it is set from the WebDAV's root directory. Otherwise it's set in the current directory.

Usage

```
wd_setwd(req, directory)
```

Arguments

req	request handle obtained from wd_connect
directory	WebDAV directory

Details

If the directory does not exist, then the request is not modified.

Notice: One has to (re)assign the returned request, as it is not modified in place.

Value

modified request

Examples

```
## Not run:
req <- wd_setwd(req, "/maindir")
req <- wd_setwd(req, "subdir")
req <- wd_setwd(req, "/othermain")

## End(Not run)
```

---

wd_upload *Uploads a file or directory to WebDAV*

Description

Directories are uploaded recursively. If the source is a file and the target a directory, then the file is uploaded into the directory. If the target is omitted, then the file or directory name (basename) will be used.

Usage

```
wd_upload(req, source, target = "")
```

Arguments

req	request handle obtained from wd_connect
source	path to local file or directory
target	path to remote file or directory, if omitted the file or directory name will be used, if source is a file and target a directory then the file will be put into the target directory

Value

vector of uploaded files (invisibly)

Examples

```
## Not run:

wd_upload(r, "d:/data/weather", "weatherfiles")
wd_upload(r, "d:/data/abc.txt", "test/xyz.txt")
wd_upload(r, "d:/data/abc.txt", "test") # uploaded file will be test/abc.txt


## End(Not run)
```

Index

basename, *17*, *21*

ncl_baseurl, 2
ncl_shareurl, 3
ncl_shareurl_from_publicurl, 4
ncl_username_from_url, 4

ocs_child_shares *(*ocs_shares_extended*)*, 11
ocs_create_share, 5
ocs_create_share_federated *(*ocs_create_share*)*, 5
ocs_create_share_group *(*ocs_create_share*)*, 5
ocs_create_share_link *(*ocs_create_share*)*, 5
ocs_create_share_mail *(*ocs_create_share*)*, 5
ocs_create_share_user *(*ocs_create_share*)*, 5
ocs_delete_share, 8
ocs_find_users, 8
ocs_modify_share, 9
ocs_send_mail, 10
ocs_share_info, 11
ocs_shares *(*ocs_shares_extended*)*, 11
ocs_shares_extended, 11

rdav, 13
rdav-package *(*rdav*)*, 13

wd_connect, *3, 4*, *6*, *8–12*, 14, *15–21*
wd_copy, 15
wd_delete, 16
wd_dir, 16
wd_download, 17
wd_getwd, 18
wd_isdir, 18
wd_mkdir, 19
wd_move, 20
wd_setwd, 20
wd_upload, 21