# rfslib

***Release 1.0.22***

**Přemysl Šťastný**

**Aug 03, 2021**

# CONTENTS:

This is a documentation of rfslib.

To create a new development enviroment, it is recommended to create python virtual enviroment and install dependencies in requirements.txt

If you want to create a new pdf documentation, you are required to install also texlive on your system.

CHAPTER

ONE

# RFSLIB.ABSTRACT_PCONNECTION MODULE

**class** `rfslib.abstract_pconnection.`**`PConnection`**(***args*)

Bases: `abc.ABC`

**`__init__`**(***args*)

TODO

**abstract** **`_exists`**(*remote_path: str*) → bool

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **`remote_path`** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**abstract** **`_isdir`**(*remote_path: str*) → bool

Protected method which checks, whether a remote file is a directory.

**Parameters** **`remote_path`** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn't a folder. Undefined if the file doesn't exist.

**abstract** **`_lexists`**(*remote_path: str*) → bool

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **`remote_path`** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**abstract** **`_listdir`**(*remote_path: str*) → List[str]

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn't exist or isn't a folder.

**Parameters** **`remote_path`** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**abstract** **`_mkdir`**(*remote_path: str*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn't exist.

**Parameters** **`remote_path`** – A path of a new remote directory.

**abstract** **`_pull`**(*remote_path: str*, *local_path: str*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn't exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`abstract _push`(*local_path: str*, *remote_path: str*)

Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`abstract _rename`(*old_name: str*, *new_name: str*)

Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`abstract _rmdir`(*remote_path: str*)

Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`abstract _unlink`(*remote_path: str*)

Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`abstract close`()

Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

**lexists**(*remote_path*)

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**listdir**(*remote_path*)

**ls**(*remote_path*)

**mkdir**(*remote_path*)

**mv**(*old_names*, *new_name*)

**pmkdir**(*remote_path*)

**pull**(*remote_path*, *local_path*)

**push**(*local_path*, *remote_path*)

**rename**(*old_name*, *new_name*)

**rm**(*remote_path*, *recursive=False*)

**rmdir**(*remote_path*)

**rpull**(*remote_path*, *local_path*)

**rpush**(*local_path*, *remote_path*)

**touch**(*remote_path*)

**unlink**(*remote_path*)

**xls**(*remote_path*)

CHAPTER

TWO

# RFSLIB.SFTP_PCONNECTION MODULE

**class** rfslib.sftp_pconnection.**SftpPConnection**(***arg*)

Bases: *rfslib.abstract_pconnection.PConnection*

**__init__**(***arg*)

TODO

**_exists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_isdir**(*remote_path*)

Protected method which checks, whether a remote file is a directory.

**Parameters** **remote_path** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn't a folder. Undefined if the file doesn't exist.

**_lexists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_listdir**(*remote_path*)

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn't exist or isn't a folder.

**Parameters** **remote_path** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**_mkdir**(*remote_path*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn't exist.

**Parameters** **remote_path** – A path of a new remote directory.

**_pull**(*remote_path*, *local_path*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn't exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`_push`(*local_path*, *remote_path*)

Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`_rename`(*old_name*, *new_name*)

Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`_rmdir`(*remote_path*)

Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`_unlink`(*remote_path*)

Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`close`()

Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

`lexists`(*remote_path*)

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`listdir`(*remote_path*)

`ls`(*remote_path*)

`mkdir`(*remote_path*)

`mv`(*old_names*, *new_name*)

`pmkdir`(*remote_path*)

`pull`(*remote_path*, *local_path*)

`push`(*local_path*, *remote_path*)

`rename`(*old_name*, *new_name*)

`rm`(*remote_path*, *recursive=False*)

`rmdir`(*remote_path*)

`rpull`(*remote_path*, *local_path*)

`rpush`(*local_path*, *remote_path*)

`touch`(*remote_path*)

`unlink`(*remote_path*)

`xls`(*remote_path*)

CHAPTER

THREE

# RFSLIB.FTP_PCONNECTION MODULE

**class** rfslib.ftp_pconnection.**FtpPConnection**(***arg*)

Bases: `rfslib.abstract_pconnection.PConnection`

**`__init__`**(***arg*)

TODO

**`_exists`**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **`remote_path`** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**`_isdir`**(*remote_path*)

Protected method which checks, whether a remote file is a directory.

**Parameters** **`remote_path`** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn't a folder. Undefined if the file doesn't exist.

**`_lexists`**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **`remote_path`** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**`_listdir`**(*remote_path*)

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn't exist or isn't a folder.

**Parameters** **`remote_path`** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**`_mkdir`**(*remote_path*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn't exist.

**Parameters** **`remote_path`** – A path of a new remote directory.

**`_pull`**(*remote_path*, *local_path*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn't exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`_push`(*local_path*, *remote_path*)
Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`_rename`(*old_name*, *new_name*)
Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`_rmdir`(*remote_path*)
Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`_unlink`(*remote_path*)
Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`close`()
Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool
Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

`lexists`(*remote_path*)
Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`listdir`(*remote_path*)

`ls`(*remote_path*)

`mkdir`(*remote_path*)

`mv`(*old_names*, *new_name*)

`pmkdir`(*remote_path*)

`pull`(*remote_path*, *local_path*)

`push`(*local_path*, *remote_path*)

`rename`(*old_name*, *new_name*)

`rm`(*remote_path*, *recursive=False*)

`rmdir`(*remote_path*)

`rpull`(*remote_path*, *local_path*)

`rpush`(*local_path*, *remote_path*)

`touch`(*remote_path*)

`unlink`(*remote_path*)

`xls`(*remote_path*)

CHAPTER

FOUR

# RFSLIB.SMB12_PCONNECTION MODULE

**class** rfslib.smb12_pconnection.**Smb12PConnection**(***arg*)

Bases: *rfslib.abstract_pconnection.PConnection*

**__init__**(***arg*)

TODO

**_exists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_isdir**(*remote_path*)

Protected method which checks, whether a remote file is a directory.

**Parameters** **remote_path** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn't a folder. Undefined if the file doesn't exist.

**_lexists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_listdir**(*remote_path*)

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn't exist or isn't a folder.

**Parameters** **remote_path** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**_mkdir**(*remote_path*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn't exist.

**Parameters** **remote_path** – A path of a new remote directory.

**_pull**(*remote_path*, *local_path*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn't exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`_push`(*local_path*, *remote_path*)

Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`_rename`(*old_name*, *new_name*)

Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`_rmdir`(*remote_path*)

Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`_unlink`(*remote_path*)

Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`close`()

Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

`lexists`(*remote_path*)
Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`listdir`(*remote_path*)

`ls`(*remote_path*)

`mkdir`(*remote_path*)

`mv`(*old_names*, *new_name*)

`pmkdir`(*remote_path*)

`pull`(*remote_path*, *local_path*)

`push`(*local_path*, *remote_path*)

`rename`(*old_name*, *new_name*)

`rm`(*remote_path*, *recursive=False*)

`rmdir`(*remote_path*)

`rpull`(*remote_path*, *local_path*)

`rpush`(*local_path*, *remote_path*)

`touch`(*remote_path*)

`unlink`(*remote_path*)

`xls`(*remote_path*)

CHAPTER

FIVE

# RFSLIB.SMB23_PCONNECTION MODULE

**class** rfslib.smb23_pconnection.**Smb23PConnection**(***arg*)

Bases: *rfslib.abstract_pconnection.PConnection*

**__init__**(***arg*)

TODO

**_exists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_isdir**(*remote_path*)

Protected method which checks, whether a remote file is a directory.

**Parameters** **remote_path** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn't a folder. Undefined if the file doesn't exist.

**_lexists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

**_listdir**(*remote_path*)

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn't exist or isn't a folder.

**Parameters** **remote_path** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**_mkdir**(*remote_path*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn't exist.

**Parameters** **remote_path** – A path of a new remote directory.

**_pull**(*remote_path*, *local_path*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn't exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`_push`(*local_path*, *remote_path*)
Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`_rename`(*old_name*, *new_name*)
Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`_rmdir`(*remote_path*)
Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`_unlink`(*remote_path*)
Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`close`()
Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool
Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

`lexists`(*remote_path*)

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`listdir`(*remote_path*)

`ls`(*remote_path*)

`mkdir`(*remote_path*)

`mv`(*old_names*, *new_name*)

`pmkdir`(*remote_path*)

`pull`(*remote_path*, *local_path*)

`push`(*local_path*, *remote_path*)

`rename`(*old_name*, *new_name*)

`rm`(*remote_path*, *recursive=False*)

`rmdir`(*remote_path*)

`rpull`(*remote_path*, *local_path*)

`rpush`(*local_path*, *remote_path*)

`touch`(*remote_path*)

`unlink`(*remote_path*)

`xls`(*remote_path*)

`rfslib.smb23_pconnection.config_smb23`(***arg*)

CHAPTER

SIX

# RFSLIB.FS_PCONNECTION MODULE

**class** rfslib.fs_pconnection.**FsPConnection**(***arg*)

Bases: *rfslib.abstract_pconnection.PConnection*

**__init__**(***arg*)

TODO

**_exists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn’t exist

**_isdir**(*remote_path*)

Protected method which checks, whether a remote file is a directory.

**Parameters** **remote_path** – A path of a directory.

**Returns** True, if remote file is folder. False, if it isn’t a folder. Undefined if the file doesn’t exist.

**_lexists**(*remote_path*)

Protected method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** **remote_path** – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn’t exist

**_listdir**(*remote_path*)

Protected method which returns a list of files in the folder including hidden files. Undefined if the remote file doesn’t exist or isn’t a folder.

**Parameters** **remote_path** – The remote path of a remote folder.

**Returns** List of files in the remote folder

**_mkdir**(*remote_path*)

Protected method which creates a new directory. Behavior is undefined if remote folder already exist, or destination folder doesn’t exist.

**Parameters** **remote_path** – A path of a new remote directory.

**_pull**(*remote_path*, *local_path*)

Protected method which downloads/pulls a file from a remote storage to a local storage in the binary form. Behavior is undefined if destination folder doesn’t exist.

**Parameters**

- `remote_path` – Path of a remote file to download.
- `local_path` – Path of a local file, where to download/pull a remote file or local file already exists.

`_push`(*local_path*, *remote_path*)

Protected method which uploads/pushes a file from a local storage to a remote storage in the binary form. Behavior is undefined if destination folder doesn't exist or remote file already exists.

**Parameters**

- `local_path` – Path of a local file to upload.
- `remote_path` – Path on the remote storage, where to upload/push a local file.

`_rename`(*old_name*, *new_name*)

Protected method which renames/moves a file. Behavior is undefined, if *new_name* file exists or *old_name* file doesn't exist.

**Parameters**

- `old_name` – Remote path a file to move.
- `new_name` – Remote path to which move the file.

`_rmdir`(*remote_path*)

Protected method which removes an empty remote directory. Behavior is undefined if remote directory doesn't exist or it isn't empty.

**Parameters** `remote_path` – Path of an empty remote directory to delete.

`_unlink`(*remote_path*)

Protected method which removes a nondirectory file. Behavior is undefined if remote file doesn't exist or is a directory.

**Parameters** `remote_path` – Path of a remote regular file to delete.

`close`()

Method to close the opened connection.

`cp`(*old_names*, *new_name*, *recursive=False*)

`dcp`(*old_names*, *target_dir*, *recursive=False*)

`dmv`(*old_names*, *target_dir*)

`exists`(*remote_path: str*) → bool

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`fcp`(*old_name*, *new_name*)

`find`(*remote_path*, *child_first=False*)

`fmv`(*old_name*, *new_name*)

`isdir`(*remote_path*)

`lexists`(*remote_path*)

Method which checks, whether a remote file exist.

KNOWN BUG: Behavior is undefined in case of broken symlinks.

**Parameters** `remote_path` – Path of a remote file.

**Returns** True, if remote file is exist. False, if remote file doesn't exist

`listdir`(*remote_path*)

`ls`(*remote_path*)

`mkdir`(*remote_path*)

`mv`(*old_names*, *new_name*)

`pmkdir`(*remote_path*)

`pull`(*remote_path*, *local_path*)

`push`(*local_path*, *remote_path*)

`rename`(*old_name*, *new_name*)

`rm`(*remote_path*, *recursive=False*)

`rmdir`(*remote_path*)

`rpull`(*remote_path*, *local_path*)

`rpush`(*local_path*, *remote_path*)

`touch`(*remote_path*)

`unlink`(*remote_path*)

`xls`(*remote_path*)

# CHAPTER SEVEN

# RFSLIB.PATH_UTILS MODULE

**class** rfslib.path_utils.**GenericPath**(*path*)

Bases: object

**__init__**(*path*)

rfslib.path_utils.**add_r_prefix**(*path*)

rfslib.path_utils.**generic_cp**(*conn*, *sources*, *dest*, *recursive=False*)

rfslib.path_utils.**generic_mv**(*conn*, *sources*, *dest*)

rfslib.path_utils.**generic_path_normalize**(*path*)

rfslib.path_utils.**is_remote**(*path*)

rfslib.path_utils.**path_normalize**(*path*)

rfslib.path_utils.**remove_r_prefix**(*path*)

# PYTHON MODULE INDEX

# INDEX