YaDisk Documentation

Release 1.2.14

Ivan Konovalov

Jul 10, 2021

Contents:

1 Introduction **1**
- 1.1 Installation . . . 1
- 1.2 Examples . . . 1

2 Documentation **5**
- 2.1 General parameters . . . 20
- 2.2 Settings . . . 20
- 2.3 Exceptions . . . 20
- 2.4 Objects . . . 23
- 2.5 Low-level API . . . 31

3 Changelog **59**

4 Donations **63**

5 Indices and tables **65**

Python Module Index **67**

Index **69**

CHAPTER 1

Introduction

YaDisk is a Yandex.Disk REST API client library.

1.1 Installation

```
pip install yadisk
```

or

```
python setup.py install
```

1.2 Examples

```
import yadisk

y = yadisk.YaDisk(token="<token>")
# or
# y = yadisk.YaDisk("<application-id>", "<application-secret>", "<token>")

# Check if the token is valid
print(y.check_token())

# Get disk information
print(y.get_disk_info())

# Print files and directories at "/some/path"
print(list(y.listdir("/some/path")))

# Upload "file_to_upload.txt" to "/destination.txt"
y.upload("file_to_upload.txt", "/destination.txt")
```

(continues on next page)

(continued from previous page)

```
# Same thing
with open("file_to_upload.txt", "rb") as f:
    y.upload(f, "/destination.txt")

# Download "/some-file-to-download.txt" to "downloaded.txt"
y.download("/some-file-to-download.txt", "downloaded.txt")

# Permanently remove "/file-to-remove"
y.remove("/file-to-remove", permanently=True)

# Create a new directory at "/test-dir"
print(y.mkdir("/test-dir"))
```

1.2.1 Receiving token with confirmation code

```
import sys
import yadisk

y = yadisk.YaDisk("application-id>", "<application-secret>")
url = y.get_code_url()

print("Go to the following url: %s" % url)
code = input("Enter the confirmation code: ")

try:
    response = y.get_token(code)
except yadisk.exceptions.BadRequestError:
    print("Bad code")
    sys.exit(1)

y.token = response.access_token

if y.check_token():
    print("Sucessfully received token!")
else:
    print("Something went wrong. Not sure how though...")
```

1.2.2 Recursive upload

```
import posixpath
import os
import yadisk

def recursive_upload(y, from_dir, to_dir):
     for root, dirs, files in os.walk(from_dir):
         p = root.split(from_dir)[1].strip(os.path.sep)
         dir_path = posixpath.join(to_dir, p)

         try:
             y.mkdir(dir_path)
         except yadisk.exceptions.PathExistsError:
```

(continues on next page)

(continued from previous page)

```
            pass

        for file in files:
            file_path = posixpath.join(dir_path, file)
            p_sys = p.replace("/", os.path.sep)
            in_path = os.path.join(from_dir, p_sys, file)
            try:
                y.upload(in_path, file_path)
            except yadisk.exceptions.PathExistsError:
                pass

y = yadisk.YaDisk(token="<application-token>")
to_dir = "/test"
from_dir = "/home/ubuntu"
recursive_upload(y, from_dir, to_dir)
```

1.2.3 Setting custom properties of files

```
import yadisk

y = yadisk.YaDisk(token="<application-token>")

path = input("Enter a path to patch: ")
properties = {"speed_of_light":       299792458,
              "speed_of_light_units": "meters per second",
              "message_for_owner":    "MWAHAHA! Your file has been patched by an evil␣
↪script!"}

meta = y.patch(path, properties)
print("\nNew properties: ")

for k, v in meta.custom_properties.items():
    print("%s: %r" % (k, v))

answer = input("\nWant to get rid of them? (y/[n]) ")

if answer.lower() in ("y", "yes"):
    properties = {k: None for k in properties}
    y.patch(path, properties)
    print("Everything's back as usual")
```

1.2.4 Emptying the trash bin

```
import sys
import time
import yadisk

y = yadisk.YaDisk(token="<application-token>")

answer = input("Are you sure about this? (y/[n]) ")

if answer.lower() in ("y", "yes"):
```

(continues on next page)

(continued from previous page)

```
    print("Emptying the trash bin...")
    operation = y.remove_trash("/")
    print("It might take a while...")

    if operation is None:
        print("Nevermind. The deed is done.")
        sys.exit(0)

    while True:
        status = y.get_operation_status(operation.href)

        if status == "in-progress":
            time.sleep(5)
            print("Still waiting...")
        elif status == "success":
            print("Success!")
            break
        else:
            print("Got some weird status: %r" % (status,))
            print("That's not normal")
            break
else:
    print("Not going to do anything")
```

CHAPTER 2

Documentation

class yadisk.**YaDisk**(*id=''*, *secret=''*, *token=''*)

Implements access to Yandex.Disk REST API.

Parameters

- **id** – application ID
- **secret** – application secret password
- **token** – application token

Variables

- **id** – *str*, application ID
- **secret** – *str*, application secret password
- **token** – *str*, application token

check_token(*token=None*, ***kwargs*)

Check whether the token is valid.

Parameters

- **token** – token to check, equivalent to *self.token* if *None*
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

clear_session_cache()

Clears the session cache. Unused sessions will be closed.

copy(*src_path*, *dst_path*, ***kwargs*)
Copy *src_path* to *dst_path*. If the operation is performed asynchronously, returns the link to the operation, otherwise, returns the link to the newly created resource.

Parameters

- **src_path** – source path
- **dst_path** – destination path
- **overwrite** – if *True* the destination path can be overwritten, otherwise, an error will be raised
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns `LinkObject` or `OperationLinkObject`

download(*src_path*, *path_or_file*, ***kwargs*)
Download the file.

Parameters

- **src_path** – source path
- **path_or_file** – destination path or file-like object
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

download_public(*public_key*, *file_or_path*, ***kwargs*)
Download the public resource.

Parameters

- **public_key** – public key or public URL of the resource
- **file_or_path** – destination path or file-like object
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

exists(*path*, ***kwargs*)
Check whether *path* exists.

Parameters

- **path** – path to the resource

- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

get_auth_url(***kwargs*)
Get authentication URL for the user to go to.

Parameters

- **type** – response type ("code" to get the confirmation code or "token" to get the token automatically)
- **device_id** – unique device ID, must be between 6 and 50 characters
- **device_name** – device name, should not be longer than 100 characters
- **display** – indicates whether to use lightweight layout, values other than "popup" are ignored
- **login_hint** – username or email for the account the token is being requested for
- **scope** – list of permissions for the application
- **optional_scope** – list of optional permissions for the application
- **force_confirm** – if True, user will be required to confirm access to the account even if the user has already granted access for the application
- **state** – The state string, which Yandex.OAuth returns without any changes (<= 1024 characters)

Returns authentication URL

get_code_url(***kwargs*)
Get the URL for the user to get the confirmation code. The confirmation code can later be used to get the token.

Parameters

- **device_id** – unique device ID, must be between 6 and 50 characters
- **device_name** – device name, should not be longer than 100 characters
- **display** – indicates whether to use lightweight layout, values other than "popup" are ignored
- **login_hint** – username or email for the account the token is being requested for
- **scope** – list of permissions for the application
- **optional_scope** – list of optional permissions for the application
- **force_confirm** – if True, user will be required to confirm access to the account even if the user has already granted access for the application
- **state** – The state string, which Yandex.OAuth returns without any changes (<= 1024 characters)

Returns authentication URL

get_disk_info(***kwargs*)
Get disk information.

Parameters

- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns DiskInfoObject

get_download_link(*path*, ***kwargs*)
Get a download link for a file (or a directory).

Parameters

- **path** – path to the resource
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

get_files(***kwargs*)
Get a flat list of all files (that doesn't include directories).

Parameters

- **offset** – offset from the beginning of the list
- **limit** – number of list elements to be included
- **media_type** – type of files to include in the list
- **sort** – *str*, field to be used as a key to sort children resources
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of ResourceObject

get_last_uploaded(***kwargs*)
Get the list of latest uploaded files sorted by upload date.

Parameters

- **limit** – maximum number of elements in the list
- **media_type** – type of files to include in the list

- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *LastUploadedResourceListObject*

get_meta(*path*, ***kwargs*)
Get meta information about a file/directory.

Parameters

- **path** – path to the resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *ResourceObject*

get_operation_status(*operation_id*, ***kwargs*)
Get operation status.

Parameters

- **operation_id** – ID of the operation or a link
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

get_public_download_link(*public_key*, ***kwargs*)
Get a download link for a public resource.

Parameters

- **public_key** – public key or public URL of the resource

- **path** – relative path to the resource within the public folder
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

get_public_meta(*public_key*, ***kwargs*)
Get meta-information about a public resource.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to a resource in a public folder. By specifying the key of the published folder in *public_key*, you can request metainformation for any resource in the folder.
- **offset** – offset from the beginning of the list of nested resources
- **limit** – maximum number of nested elements to be included in the list
- **sort** – *str*, field to be used as a key to sort children resources
- **preview_size** – file preview size
- **preview_crop** – *bool*, allow preview crop
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *PublicResourceObject*

get_public_resources(***kwargs*)
Get a list of public resources.

Parameters

- **offset** – offset from the beginning of the list
- **limit** – maximum number of elements in the list
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **type** – filter based on type of resources ("file" or "dir")
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *PublicResourcesListObject*

get_public_type(*public_key*, ***kwargs*)
Get public resource type.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

get_session(*token=None*)
Like *YaDisk.make_session* but wrapped in `functools.lru_cache`.

Returns `requests.Session`, different instances for different threads

get_token(*code*, ***kwargs*)
Get a new token.

Parameters

- **code** – confirmation code
- **device_id** – unique device ID (between 6 and 50 characters)
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenObject*

get_trash_meta(*path*, ***kwargs*)
Get meta information about a trash resource.

Parameters

- **path** – path to the trash resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries

- **retry_interval** – delay between retries in seconds

Returns *TrashResourceObject*

get_trash_type(*path*, ***kwargs*)
Get trash resource type.

Parameters

- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

get_type(*path*, ***kwargs*)
Get resource type.

Parameters

- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

get_upload_link(*path*, ***kwargs*)
Get a link to upload the file using the PUT request.

Parameters

- **path** – destination path
- **overwrite** – *bool*, determines whether to overwrite the destination
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

is_dir(*path*, ***kwargs*)
Check whether *path* is a directory.

Parameters

- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers

- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a directory, *False* otherwise (even if it doesn't exist)

is_file(*path*, ***kwargs*)
Check whether *path* is a file.

Parameters

- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a file, *False* otherwise (even if it doesn't exist)

is_public_dir(*public_key*, ***kwargs*)
Check whether *public_key* is a public directory.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *public_key* is a directory, *False* otherwise (even if it doesn't exist)

is_public_file(*public_key*, ***kwargs*)
Check whether *public_key* is a public file.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *public_key* is a file, *False* otherwise (even if it doesn't exist)

is_trash_dir(*path*, ***kwargs*)
Check whether *path* is a trash directory.

Parameters

- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout

- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a directory, *False* otherwise (even if it doesn't exist)

is_trash_file(*path*, ***kwargs*)
Check whether *path* is a trash file.

Parameters

- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a directory, *False* otherwise (even if it doesn't exist)

listdir(*path*, ***kwargs*)
Get contents of *path*.

Parameters

- **path** – path to the directory
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *ResourceObject*

make_session(*token=None*)
Prepares `requests.Session` object with headers needed for API.

Parameters **token** – application token, equivalent to *self.token* if *None*

Returns `requests.Session`

mkdir(*path*, ***kwargs*)
Create a new directory.

Parameters

- **path** – path to the directory to be created
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout

- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*

move(*src_path*, *dst_path*, ***kwargs*)
Move *src_path* to *dst_path*.

Parameters

- **src_path** – source path to be moved
- **dst_path** – destination path
- **overwrite** – *bool*, determines whether to overwrite the destination
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *OperationLinkObject* or *LinkObject*

patch(*path*, *properties*, ***kwargs*)
Update custom properties of a resource.

Parameters

- **path** – path to the resource
- **properties** – *dict*, custom properties to update
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *ResourceObject*

public_exists(*public_key*, ***kwargs*)
Check whether the public resource exists.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

public_listdir(*public_key*, ***kwargs*)
Get contents of a public directory.

Parameters

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource in the public folder. By specifying the key of the published folder in *public_key*, you can request contents of any nested folder.
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *PublicResourceObject*

publish(*path*, ***kwargs*)
Make a resource public.

Parameters

- **path** – path to the resource to be published
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*, link to the resource

refresh_token(*refresh_token*, ***kwargs*)
Refresh an existing token.

Parameters

- **refresh_token** – the refresh token that was received with the token
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenObject*

remove(*path*, ***kwargs*)
Remove the resource.

Parameters

- **path** – path to the resource to be removed
- **permanently** – if *True*, the resource will be removed permanently, otherwise, it will be just moved to the trash
- **md5** – *str*, MD5 hash of the file to remove
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns `OperationLinkObject` if the operation is performed asynchronously, *None* otherwise

remove_trash(*path*, ***kwargs*)

Remove a trash resource.

Parameters

- **path** – path to the trash resource to be deleted
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns `OperationLinkObject` if the operation is performed asynchronously, *None* otherwise

restore_trash(*path*, *dst_path=None*, ***kwargs*)

Restore a trash resource. Returns a link to the newly created resource or a link to the asynchronous operation.

Parameters

- **path** – path to the trash resource to restore
- **dst_path** – destination path
- **overwrite** – *bool*, determines whether the destination can be overwritten
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

revoke_token(*token=None*, ***kwargs*)
Revoke the token.

Parameters

- **token** – token to revoke, equivalent to *self.token* if *None*
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenRevokeStatusObject*

save_to_disk(*public_key*, ***kwargs*)
Saves a public resource to the disk. Returns the link to the operation if it's performed asynchronously, or a link to the resource otherwise.

Parameters

- **public_key** – public key or public URL of the resource
- **name** – filename of the saved resource
- **path** – path to the copied resource in the public folder
- **save_path** – path to the destination directory (downloads directory by default)
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

trash_exists(*path*, ***kwargs*)
Check whether the trash resource at *path* exists.

Parameters

- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

trash_listdir(*path*, ***kwargs*)
Get contents of a trash resource.

Parameters

- **path** – path to the directory in the trash bin

- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *TrashResourceObject*

unpublish(*path*, ***kwargs*)
Make a public resource private.

Parameters

- **path** – path to the resource to be unpublished
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*, link to the resource

upload(*path_or_file*, *dst_path*, ***kwargs*)
Upload a file to disk.

Parameters

- **path_or_file** – path or file-like object to be uploaded
- **dst_path** – destination path
- **overwrite** – if *True*, the resource will be overwritten if it already exists, an error will be raised otherwise
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

upload_url(*url*, *path*, ***kwargs*)
Upload a file from URL.

Parameters

- **url** – source URL
- **path** – destination path

- **disable_redirects** – *bool*, forbid redirects
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns `OperationLinkObject`, link to the asynchronous operation

2.1 General parameters

Almost all methods of *YaDisk* (the ones that accept ***kwargs*) accept some additional arguments:

- **n_retries** - *int*, maximum number of retries for a request
- **retry_delay** - *float*, delay between retries (in seconds)
- **headers** - *dict* or *None*, additional request headers

requests parameters like *timeout*, *proxies*, etc. are also accepted (see `requests.request()`).

This also applies to low-level functions and API request objects as well.

2.2 Settings

The following settings can be accessed and changed at runtime in *yadisk.settings* module:

- **DEFAULT_TIMEOUT** - *tuple* of 2 numbers (*int* or float‘), default timeout for requests. First number is the connect timeout, the second one is the read timeout.
- **DEFAULT_N_RETRIES** - *int*, default number of retries
- **DEFAULT_RETRY_INTERVAL** - *float*, default retry interval
- **DEFAULT_UPLOAD_TIMEOUT** - analogous to *DEFAULT_TIMEOUT* but for *upload* function
- **DEFAULT_UPLOAD_RETRY_INTERVAL** - analogous to *DEFAULT_RETRY_INTERVAL* but for *upload* function

2.3 Exceptions

Aside from the exceptions listed below, API requests can also raise exceptions in *requests*.

exception yadisk.exceptions.**YaDiskError**(*error_type=None*, *msg=''*, *response=None*)

Bases: `Exception`

Base class for all exceptions in this library.

Variables

- **error_type** – *str*, unique error code as returned by API
- **response** – an instance of `requests.Response`

Parameters

- **error_type** – *str*, unique error code as returned by API
- **msg** – *str*, exception message
- **response** – an instance of `requests.Response`

exception yadisk.exceptions.**RetriableYaDiskError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when there was an error but it would make sense to retry the request.

exception yadisk.exceptions.**UnknownYaDiskError**(*msg=''*, *response=None*)
Bases: *yadisk.exceptions.RetriableYaDiskError*

Thrown when the request failed but the response does not contain any error info.

exception yadisk.exceptions.**WrongResourceTypeError**(*msg=''*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the resource was expected to be of different type (e.g., file instead of directory).

exception yadisk.exceptions.**BadRequestError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 400.

exception yadisk.exceptions.**UnauthorizedError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 401.

exception yadisk.exceptions.**ForbiddenError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 403.

exception yadisk.exceptions.**NotFoundError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 404.

exception yadisk.exceptions.**NotAcceptableError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 406.

exception yadisk.exceptions.**ConflictError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 409.

exception yadisk.exceptions.**UnsupportedMediaError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 415.

exception yadisk.exceptions.**LockedError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 423.

exception yadisk.exceptions.**TooManyRequestsError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 429.

exception yadisk.exceptions.**InternalServerError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.RetriableYaDiskError*

Thrown when the server returns code 500.

exception yadisk.exceptions.**BadGatewayError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.RetriableYaDiskError*

Thrown when the server returns code 502

exception yadisk.exceptions.**UnavailableError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.RetriableYaDiskError*

Thrown when the server returns code 503.

exception yadisk.exceptions.**GatewayTimeoutError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.RetriableYaDiskError*

Thrown when the server returns code 504

exception yadisk.exceptions.**InsufficientStorageError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.YaDiskError*

Thrown when the server returns code 509.

exception yadisk.exceptions.**PathNotFoundError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.NotFoundError*

Thrown when the requested path does not exist.

exception yadisk.exceptions.**ParentNotFoundError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.ConflictError*

Thrown by *mkdir*, *upload*, etc. when the parent directory doesn't exist.

exception yadisk.exceptions.**PathExistsError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.ConflictError*

Thrown when the requested path already exists.

exception yadisk.exceptions.**DirectoryExistsError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.PathExistsError*

Thrown when the directory already exists.

exception yadisk.exceptions.**FieldValidationError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.BadRequestError*

Thrown when the request contains fields with invalid data.

exception yadisk.exceptions.**ResourceIsLockedError**(*error_type=None*, *msg=''*, *response=None*)

Bases: *yadisk.exceptions.LockedError*

Thrown when the resource is locked by another operation.

exception yadisk.exceptions.**MD5DifferError**(*error_type=None*, *msg=''*, *response=None*)
Bases: *yadisk.exceptions.ConflictError*

Thrown when the MD5 hash of the file to be deleted doesn't match with the actual one.

2.4 Objects

class yadisk.objects.**YaDiskObject**(*field_types=None*)
Base class for all objects mirroring the ones returned by Yandex.Disk REST API. It must have a fixed number of fields, each field must have a type. It also supports subscripting and access of fields through the . operator.

Parameters **field_types** – *dict* or *None*

import_fields(*source_dict*)
Set all the fields of the object to the values in *source_dict*. All the other fields are ignored

Parameters **source_dict** – *dict* or *None* (nothing will be done in that case)

remove_alias(*alias*)
Remove an alias.

Parameters **alias** – *str*

remove_field(*field*)
Remove field.

Parameters **field** – *str*

set_alias(*alias*, *name*)
Set an alias.

Parameters

- **alias** – *str*, alias to add
- **name** – *str*, field name

set_field_type(*field*, *type*)
Set field type.

Parameters

- **field** – *str*
- **type** – type or factory

set_field_types(*field_types*)
Set the field types of the object

Parameters **field_types** – *dict*, where keys are the field names and values are types (or factories)

class yadisk.objects.**ErrorObject**(*error=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Mirrors Yandex.Disk REST API error object.

Parameters **error** – *dict* or *None*

Variables

- **message** – *str*, human-readable error message

- **description** – *str*, technical error description
- **error** – *str*, error code

class yadisk.objects.auth.**TokenObject**(*token=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Token object.

Parameters **token** – *dict* or *None*

Variables

- **access_token** – *str*, token string
- **refresh_token** – *str*, the refresh-token
- **token_type** – *str*, type of the token
- **expires_in** – *int*, amount of time before the token expires

class yadisk.objects.auth.**TokenRevokeStatusObject**(*token_revoke_status=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Result of token revocation request.

Parameters **token_revoke_status** – *dict* or *None*

Variables **status** – *str*, status of the operation

class yadisk.objects.disk.**DiskInfoObject**(*disk_info=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Disk information object.

Parameters **disk_info** – *dict* or *None*

Variables

- **max_file_size** – *int*, maximum supported file size (bytes)
- **unlimited_autoupload_enabled** – *bool*, tells whether unlimited autoupload from mobile devices is enabled
- **total_space** – *int*, total disk size (bytes)
- **trash_size** – *int*, amount of space used by trash (bytes), part of *used_space*
- **is_paid** – *bool*, tells if the account is paid or not
- **used_space** – *int*, amount of space used (bytes)
- **system_folders** – *SystemFoldersObject*, paths to the system folders
- **user** – *UserObject*, owner of the disk
- **revision** – *int*, current revision of Yandex.Disk

class yadisk.objects.disk.**SystemFoldersObject**(*system_folders=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Object, containing paths to system folders.

Parameters **system_folders** – *dict* or *None*

Variables

- **odnoklassniki** – *str*, path to the Odnoklassniki folder
- **google** – *str*, path to the Google+ folder

- **instagram** – *str*, path to the Instagram folder
- **vkontakte** – *str*, path to the VKontakte folder
- **mailru** – *str*, path to the My World folder
- **facebook** – *str*, path to the Facebook folder
- **social** – *str*, path to the social networks folder
- **screenshots** – *str*, path to the screenshot folder
- **photostream** – *str*, path to the camera folder

class yadisk.objects.disk.**UserObject**(*user=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

User object.

Parameters **user** – *dict* or *None*

Variables

- **country** – *str*, user's country
- **login** – *str*, user's login
- **display_name** – *str*, user's display name
- **uid** – *str*, user's UID

class yadisk.objects.disk.**UserPublicInfoObject**(*public_user_info=None*)
Bases: *yadisk.objects.disk.UserObject*

Public user information object. Inherits from *UserObject* for compatibility.

Parameters **public_user_info** – *dict* or *None*

Variables

- **login** – *str*, user's login
- **display_name** – *str*, user's display name
- **uid** – *str*, user's UID

class yadisk.objects.resources.**CommentIDsObject**(*comment_ids=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Comment IDs object.

Parameters **comment_ids** – *dict* or *None*

Variables

- **private_resource** – *str*, comment ID for private resources
- **public_resource** – *str*, comment ID for public resources

class yadisk.objects.resources.**EXIFObject**(*exif=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

EXIF metadata object.

Parameters **exif** – *dict* or *None*

Variables **date_time** – datetime.datetime, capture date

class yadisk.objects.resources.**FilesResourceListObject**(*files_resource_list=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Flat list of files.

Parameters **files_resource_list** – *dict* or *None*

Variables

- **items** – *list*, flat list of files (*ResourceObject*)
- **limit** – *int*, maximum number of elements in the list
- **offset** – *int*, offset from the beginning of the list

class yadisk.objects.resources.**LastUploadedResourceListObject**(*last_uploaded_resources_list=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

List of last uploaded resources.

Parameters **last_uploaded_resources_list** – *dict* or *None*

Variables

- **items** – *list*, list of resources (*ResourceObject*)
- **limit** – *int*, maximum number of elements in the list

class yadisk.objects.resources.**LinkObject**(*link=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Link object.

Parameters **link** – *dict* or *None*

Variables

- **href** – *str*, link URL
- **method** – *str*, HTTP method
- **templated** – *bool*, tells whether the URL is templated

class yadisk.objects.resources.**OperationLinkObject**(*link=None*)
Bases: *yadisk.objects.resources.LinkObject*

Operation link object.

Parameters **link** – *dict* or *None*

Variables

- **href** – *str*, link URL
- **method** – *str*, HTTP method
- **templated** – *bool*, tells whether the URL is templated

class yadisk.objects.resources.**PublicResourcesListObject**(*public_resources_list=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

List of public resources.

Parameters **public_resources_list** – *dict* or *None*

Variables

- **items** – *list*, list of public resources (*PublicResourceObject*)
- **type** – *str*, resource type to filter by

- **limit** – *int*, maximum number of elements in the list
- **offset** – *int*, offset from the beginning of the list

class yadisk.objects.resources.**ResourceListObject**(*resource_list=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

List of resources.

Parameters **resource_list** – *dict* or *None*

Variables

- **sort** – *str*, sort type
- **items** – *list*, list of resources (*ResourceObject*)
- **limit** – *int*, maximum number of elements in the list
- **offset** – *int*, offset from the beginning of the list
- **path** – *str*, path to the directory that contains the elements of the list
- **total** – *int*, number of elements in the list

class yadisk.objects.resources.**ResourceObject**(*resource=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Resource object.

Parameters **resource** – *dict* or *None*

Variables

- **antivirus_status** – *str*, antivirus check status
- **file** – *str*, download URL
- **size** – *int*, file size
- **public_key** – *str*, public resource key
- **sha256** – *str*, SHA256 hash
- **md5** – *str*, MD5 hash
- **embedded** – *ResourceListObject*, list of nested resources
- **name** – *str*, filename
- **exif** – *EXIFObject*, EXIF metadata
- **resource_id** – *str*, resource ID
- **custom_properties** – *dict*, custom resource properties
- **public_url** – *str*, public URL
- **share** – *ShareInfoObject*, shared folder information
- **modified** – datetime.datetime, date of last modification
- **created** – datetime.datetime, date of creation
- **photoslice_time** – datetime.datetime, photo/video creation date
- **mime_type** – *str*, MIME type
- **path** – *str*, path to the resource
- **preview** – *str*, file preview URL

- **comment_ids** – *CommentIDsObject*, comment IDs
- **type** – *str*, type ("file" or "dir")
- **media_type** – *str*, file type as determined by Yandex.Disk
- **revision** – *int*, Yandex.Disk revision at the time of last modification

class yadisk.objects.resources.**ResourceUploadLinkObject**(*resource_upload_link=None*)
Bases: *yadisk.objects.resources.LinkObject*

Resource upload link.

Parameters **resource_upload_link** – *dict* or *None*

Variables

- **operation_id** – *str*, ID of the upload operation
- **href** – *str*, link URL
- **method** – *str*, HTTP method
- **templated** – *bool*, tells whether the URL is templated

class yadisk.objects.resources.**ShareInfoObject**(*share_info=None*)
Bases: yadisk.objects.yadisk_object.YaDiskObject

Shared folder information object.

Parameters **share_info** – *dict* or *None*

Variables

- **is_root** – *bool*, tells whether the folder is root
- **is_owned** – *bool*, tells whether the user is the owner of this directory
- **rights** – *str*, access rights

class yadisk.objects.resources.**PublicResourceObject**(*public_resource=None*)
Bases: *yadisk.objects.resources.ResourceObject*

Public resource object.

Parameters **resource** – *dict* or *None*

Variables

- **antivirus_status** – *str*, antivirus check status
- **file** – *str*, download URL
- **size** – *int*, file size
- **public_key** – *str*, public resource key
- **sha256** – *str*, SHA256 hash
- **md5** – *str*, MD5 hash
- **embedded** – *PublicResourceObject*, list of nested resources
- **name** – *str*, filename
- **exif** – *EXIFObject*, EXIF metadata
- **resource_id** – *str*, resource ID
- **custom_properties** – *dict*, custom resource properties

- **public_url** – *str*, public URL
- **share** – *ShareInfoObject*, shared folder information
- **modified** – datetime.datetime, date of last modification
- **created** – datetime.datetime, date of creation
- **photoslice_time** – datetime.datetime, photo/video creation date
- **mime_type** – *str*, MIME type
- **path** – *str*, path to the resource
- **preview** – *str*, file preview URL
- **comment_ids** – *CommentIDsObject*, comment IDs
- **type** – *str*, type ("file" or "dir")
- **media_type** – *str*, file type as determined by Yandex.Disk
- **revision** – *int*, Yandex.Disk revision at the time of last modification
- **view_count** – *int*, number of times the public resource was viewed
- **owner** – *UserPublicInfoObject*, owner of the public resource

class yadisk.objects.resources.**PublicResourceListObject**(*public_resource_list=None*)
Bases: *yadisk.objects.resources.ResourceListObject*

List of public resources.

Parameters **public_resource_list** – *dict* or *None*

Variables

- **sort** – *str*, sort type
- **items** – *list*, list of resources (*ResourceObject*)
- **limit** – *int*, maximum number of elements in the list
- **offset** – *int*, offset from the beginning of the list
- **path** – *str*, path to the directory that contains the elements of the list
- **total** – *int*, number of elements in the list
- **public_key** – *str*, public key of the resource

class yadisk.objects.resources.**TrashResourceObject**(*trash_resource=None*)
Bases: *yadisk.objects.resources.ResourceObject*

Trash resource object.

Parameters **trash_resource** – *dict* or *None*

Variables

- **antivirus_status** – *str*, antivirus check status
- **file** – *str*, download URL
- **size** – *int*, file size
- **public_key** – *str*, public resource key
- **sha256** – *str*, SHA256 hash
- **md5** – *str*, MD5 hash

- **embedded** – *TrashResourceListObject*, list of nested resources
- **name** – *str*, filename
- **exif** – *EXIFObject*, EXIF metadata
- **resource_id** – *str*, resource ID
- **custom_properties** – *dict*, custom resource properties
- **public_url** – *str*, public URL
- **share** – *ShareInfoObject*, shared folder information
- **modified** – datetime.datetime, date of last modification
- **created** – datetime.datetime, date of creation
- **photoslice_time** – datetime.datetime, photo/video creation date
- **mime_type** – *str*, MIME type
- **path** – *str*, path to the resource
- **preview** – *str*, file preview URL
- **comment_ids** – *CommentIDsObject*, comment IDs
- **type** – *str*, type ("file" or "dir")
- **media_type** – *str*, file type as determined by Yandex.Disk
- **revision** – *int*, Yandex.Disk revision at the time of last modification
- **origin_path** – *str*, original path
- **deleted** – datetime.datetime, date of deletion

class yadisk.objects.resources.**TrashResourceListObject**(*trash_resource_list=None*)

Bases: *yadisk.objects.resources.ResourceListObject*

List of trash resources.

Parameters **trash_resource_list** – *dict* or *None*

Variables

- **sort** – *str*, sort type
- **items** – *list*, list of resources (*TrashResourceObject*)
- **limit** – *int*, maximum number of elements in the list
- **offset** – *int*, offset from the beginning of the list
- **path** – *str*, path to the directory that contains the elements of the list
- **total** – *int*, number of elements in the list

class yadisk.objects.operations.**OperationStatusObject**(*operation_status=None*)

Bases: yadisk.objects.yadisk_object.YaDiskObject

Operation status object.

Parameters **operation_status** – *dict* or *None*

Variables

- **type** – *str*, type of the operation
- **status** – *str*, status of the operation

- **operation_id** – *str*, ID of the operation
- **link** – *LinkObject*, link to the operation
- **data** – *dict*, other information about the operation

2.5 Low-level API

2.5.1 Utilities

yadisk.utils.**get_exception**(*response*)
Get an exception instance based on response, assuming the request has failed.

Parameters **response** – an instance of `requests.Response`

Returns an exception instance, subclass of *YaDiskError*

yadisk.utils.**auto_retry**(*func*, *n_retries=None*, *retry_interval=None*)
Attempt to perform a request with automatic retries. A retry is triggered by `requests.exceptions.RequestException` or *RetriableYaDiskError*.

Parameters

- **func** – function to run, must not require any arguments
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – *int* or *float*, delay between retries (in seconds)

Returns return value of func()

2.5.2 Functions

yadisk.functions.auth.**check_token**(*session*, ***kwargs*)
Check whether the token is valid.

Parameters **session** – an instance of `requests.Session` with prepared headers

Returns *bool*

yadisk.functions.auth.**get_auth_url**(*client_id*, ***kwargs*)
Get authentication URL for the user to go to.

Parameters

- **client_id** – application ID
- **type** – response type ("code" to get the confirmation code or "token" to get the token automatically)
- **device_id** – unique device ID, must be between 6 and 50 characters
- **device_name** – device name, should not be longer than 100 characters
- **display** – indicates whether to use lightweight layout, values other than "popup" are ignored
- **login_hint** – username or email for the account the token is being requested for
- **scope** – list of permissions for the application
- **optional_scope** – list of optional permissions for the application

- **force_confirm** – if True, user will be required to confirm access to the account even if the user has already granted access for the application
- **state** – The state string, which Yandex.OAuth returns without any changes (<= 1024 characters)

Returns authentication URL

yadisk.functions.auth.**get_code_url**(*client_id*, ***kwargs*)

Get the URL for the user to get the confirmation code. The confirmation code can later be used to get the token.

Parameters

- **client_id** – application ID
- **device_id** – unique device ID, must be between 6 and 50 characters
- **device_name** – device name, should not be longer than 100 characters
- **display** – indicates whether to use lightweight layout, values other than "popup" are ignored
- **login_hint** – username or email for the account the token is being requested for
- **scope** – list of permissions for the application
- **optional_scope** – list of optional permissions for the application
- **force_confirm** – if True, user will be required to confirm access to the account even if the user has already granted access for the application
- **state** – The state string, which Yandex.OAuth returns without any changes (<= 1024 characters)

Returns authentication URL

yadisk.functions.auth.**get_token**(*code*, *client_id*, *client_secret*, ***kwargs*)

Get a new token.

Parameters

- **code** – confirmation code
- **client_id** – application ID
- **client_secret** – application secret password
- **device_id** – unique device ID (between 6 and 50 characters)
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenObject*

yadisk.functions.auth.**refresh_token**(*refresh_token*, *client_id*, *client_secret*, ***kwargs*)

Refresh an existing token.

Parameters

- **refresh_token** – the refresh token that was received with the token
- **client_id** – application ID
- **client_secret** – application secret password

- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenObject*

yadisk.functions.auth.**revoke_token**(*token*, *client_id*, *client_secret*, ***kwargs*)
Revoke the token.

Parameters

- **token** – token to revoke
- **client_id** – application ID
- **client_secret** – application secret password
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TokenRevokeStatusObject*

yadisk.functions.disk.**get_disk_info**(*session*, ***kwargs*)
Get disk information.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *DiskInfoObject*

yadisk.functions.resources.**copy**(*session*, *src_path*, *dst_path*, ***kwargs*)
Copy *src_path* to *dst_path*. If the operation is performed asynchronously, returns the link to the operation, otherwise, returns the link to the newly created resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **src_path** – source path
- **dst_path** – destination path
- **overwrite** – if *True* the destination path can be overwritten, otherwise, an error will be raised
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout

- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

yadisk.functions.resources.**download**(*session*, *src_path*, *file_or_path*, ***kwargs*)
Download the file.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **src_path** – source path
- **path_or_file** – destination path or file-like object
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

yadisk.functions.resources.**exists**(*session*, *path*, ***kwargs*)
Check whether *path* exists.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

yadisk.functions.resources.**get_download_link**(*session*, *path*, ***kwargs*)
Get a download link for a file (or a directory).

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

yadisk.functions.resources.**get_meta**(*session*, *path*, ***kwargs*)
Get meta information about a file/directory.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *ResourceObject*

yadisk.functions.resources.**get_type**(*session*, *path*, ***kwargs*)
Get resource type.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

yadisk.functions.resources.**get_upload_link**(*session*, *path*, ***kwargs*)
Get a link to upload the file using the PUT request.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – destination path
- **overwrite** – *bool*, determines whether to overwrite the destination
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

yadisk.functions.resources.**is_dir**(*session*, *path*, ***kwargs*)
Check whether *path* is a directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a directory, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**is_file**(*session*, *path*, ***kwargs*)
Check whether *path* is a file.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a file, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**listdir**(*session*, *path*, ***kwargs*)
Get contents of *path*.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the directory
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *ResourceObject*

yadisk.functions.resources.**mkdir**(*session*, *path*, ***kwargs*)
Create a new directory.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the directory to be created
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*

yadisk.functions.resources.**remove**(*session*, *path*, ***kwargs*)
Remove the resource.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the resource to be removed
- **permanently** – if *True*, the resource will be removed permanently, otherwise, it will be just moved to the trash
- **md5** – *str*, MD5 hash of the file to remove
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* if the operation is performed asynchronously, *None* otherwise

yadisk.functions.resources.**upload**(*session*, *file_or_path*, *dst_path*, ***kwargs*)
Upload a file to disk.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **file_or_path** – path or file-like object to be uploaded
- **dst_path** – destination path
- **overwrite** – if *True*, the resource will be overwritten if it already exists, an error will be raised otherwise
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

yadisk.functions.resources.**get_trash_meta**(*session*, *path*, ***kwargs*)
Get meta information about a trash resource.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the trash resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *TrashResourceObject*

yadisk.functions.resources.**trash_exists**(*session*, *path*, ***kwargs*)

Check whether the trash resource at *path* exists.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

yadisk.functions.resources.**restore_trash**(*session*, *path*, *dst_path=None*, ***kwargs*)

Restore a trash resource. Returns a link to the newly created resource or a link to the asynchronous operation.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the trash resource to be restored
- **dst_path** – destination path
- **overwrite** – *bool*, determines whether the destination can be overwritten
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

yadisk.functions.resources.**move**(*session*, *src_path*, *dst_path*, ***kwargs*)
Move *src_path* to *dst_path*.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **src_path** – source path to be moved
- **dst_path** – destination path
- **overwrite** – *bool*, determines whether to overwrite the destination
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

yadisk.functions.resources.**remove_trash**(*session*, *path*, ***kwargs*)
Remove a trash resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the trash resource to be deleted
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *OperationLinkObject* if the operation is performed asynchronously, *None* otherwise

yadisk.functions.resources.**publish**(*session*, *path*, ***kwargs*)
Make a resource public.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be published
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*, link to the resource

yadisk.functions.resources.**unpublish**(*session*, *path*, ***kwargs*)
Make a public resource private.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be unpublished
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject*

yadisk.functions.resources.**save_to_disk**(*session*, *public_key*, ***kwargs*)
Saves a public resource to the disk. Returns the link to the operation if it's performed asynchronously, or a link to the resource otherwise.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **name** – filename of the saved resource
- **path** – path to the copied resource in the public folder
- **save_path** – path to the destination directory (downloads directory by default)
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *LinkObject* or *OperationLinkObject*

yadisk.functions.resources.**get_public_meta**(*session*, *public_key*, ***kwargs*)
Get meta-information about a public resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to a resource in a public folder. By specifying the key of the published folder in *public_key*, you can request metainformation for any resource in the folder.
- **offset** – offset from the beginning of the list of nested resources
- **limit** – maximum number of nested elements to be included in the list
- **sort** – *str*, field to be used as a key to sort children resources

- **preview_size** – file preview size
- **preview_crop** – *bool*, allow preview crop
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *PublicResourceObject*

yadisk.functions.resources.**public_exists**(*session*, *public_key*, ***kwargs*)
Check whether the public resource exists.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *bool*

yadisk.functions.resources.**public_listdir**(*session*, *public_key*, ***kwargs*)
Get contents of a public directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource in the public folder. By specifying the key of the published folder in *public_key*, you can request contents of any nested folder.
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *PublicResourceObject*

yadisk.functions.resources.**get_public_type**(*session*, *public_key*, ***kwargs*)
Get public resource type.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

yadisk.functions.resources.**is_public_dir**(*session*, *public_key*, ***kwargs*)
Check whether the public resource is a public directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *public_key* is a directory, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**is_public_file**(*session*, *public_key*, ***kwargs*)
Check whether the public resource is a public file.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *public_key* is a file, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**trash_listdir**(*session*, *path*, ***kwargs*)
Get contents of a trash resource.

Parameters

- **session** – an instance of requests.Session with prepared headers

- **path** – path to the directory in the trash bin
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *TrashResourceObject*

yadisk.functions.resources.**get_trash_type**(*session*, *path*, ***kwargs*)
Get trash resource type.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns "file" or "dir"

yadisk.functions.resources.**is_trash_dir**(*session*, *path*, ***kwargs*)
Check whether *path* is a trash directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a directory, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**is_trash_file**(*session*, *path*, ***kwargs*)
Check whether *path* is a trash file.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the trash resource
- **timeout** – *float* or *tuple*, request timeout

- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *True* if *path* is a file, *False* otherwise (even if it doesn't exist)

yadisk.functions.resources.**get_public_resources**(*session*, ***kwargs*)
Get a list of public resources.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **offset** – offset from the beginning of the list
- **limit** – maximum number of elements in the list
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **type** – filter based on type of resources ("file" or "dir")
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *PublicResourcesListObject*

yadisk.functions.resources.**patch**(*session*, *path*, *properties*, ***kwargs*)
Update custom properties of a resource.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the resource
- **properties** – *dict*, custom properties to update
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *ResourceObject*

yadisk.functions.resources.**get_files**(*session*, ***kwargs*)
Get a flat list of all files (that doesn't include directories).

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **offset** – offset from the beginning of the list
- **limit** – number of list elements to be included

- **media_type** – type of files to include in the list
- **sort** – *str*, field to be used as a key to sort children resources
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *ResourceObject*

yadisk.functions.resources.**get_last_uploaded**(*session*, ***kwargs*)
Get the list of latest uploaded files sorted by upload date.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **limit** – maximum number of elements in the list
- **media_type** – type of files to include in the list
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns generator of *LastUploadedResourceListObject*

yadisk.functions.resources.**upload_url**(*session*, *url*, *path*, ***kwargs*)
Upload a file from URL.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **url** – source URL
- **path** – destination path
- **disable_redirects** – *bool*, forbid redirects
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *OperationLinkObject*, link to the asynchronous operation

yadisk.functions.resources.**get_public_download_link**(*session*, *public_key*, ***kwargs*)
Get a download link for a public resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **path** – relative path to the resource within the public folder
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

yadisk.functions.resources.**download_public**(*session*, *public_key*, *file_or_path*, ***kwargs*)
Download the public resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the public resource
- **file_or_path** – destination path or file-like object
- **path** – relative path to the resource within the public folder
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

yadisk.functions.operations.**get_operation_status**(*session*, *operation_id*, ***kwargs*)
Get operation status.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **operation_id** – ID of the operation or a link
- **fields** – list of keys to be included in the response
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds

Returns *str*

2.5.3 API request objects

class yadisk.api.**APIRequest**(*session*, *args*, ***kwargs*)
Base class for all API requests.

Parameters

- **session** – an instance of requests.Session
- **args** – *dict* of arguments, that will be passed to *process_args*
- **timeout** – *float* or *tuple*, request timeout
- **headers** – *dict* or *None*, additional request headers
- **n_retries** – *int*, maximum number of retries
- **retry_interval** – delay between retries in seconds
- **kwargs** – other arguments for requests.Session.send

Variables

- **url** – *str*, request URL
- **method** – *str*, request method
- **content_type** – *str*, Content-Type header ("application/x-www-form-urlencoded" by default)
- **timeout** – *float* or *tuple*, request timeout
- **n_retries** – *int*, maximum number of retries
- **success_codes** – *list*-like, list of response codes that indicate request's success
- **retry_interval** – *float*, delay between retries in seconds

prepare()
Prepare the request

process()
Process the response.

Returns depends on *self.process_json()*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

send()

Actually send the request

Returns requests.Response (*self.response*)

class yadisk.api.auth.**RefreshTokenRequest**(*session*, *refresh_token*, *client_id*, *client_secret*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to refresh an existing token.

Parameters

- **session** – an instance of requests.Session with prepared headers

- **refresh_token** – the refresh token that was received with the original token
- **client_id** – application ID
- **client_secret** – application secret password

Returns *TokenObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.auth.**RevokeTokenRequest**(*session*, *token*, *client_id*, *client_secret*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to revoke the token.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **token** – the token to be revoked
- **client_id** – application ID
- **client_secret** – application secret password

Returns *TokenRevokeStatusObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.auth.**GetTokenRequest**(*session*, *code*, *client_id*, *client_secret*, *device_id=None*, *device_name=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get the token.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **code** – confirmation code
- **client_id** – application ID
- **client_secret** – application secret password
- **device_id** – unique device ID (between 6 and 50 characters)

Returns *TokenObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.disk.**DiskInfoRequest**(*session*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get disk information.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **fields** – list of keys to be included in the response

Returns *DiskInfoObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetPublicResourcesRequest**(*session*, *offset=0*, *limit=20*, *preview_size=None*, *preview_crop=None*, *type=None*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get a list of public resources.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **offset** – offset from the beginning of the list
- **limit** – maximum number of elements in the list
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **type** – filter based on type of resources (“file” or “dir”)
- **fields** – list of keys to be included in the response

Returns *PublicResourcesListObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**UnpublishRequest**(*session*, *path*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to make a public resource private.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be unpublished
- **fields** – list of keys to be included in the response

Returns *LinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetDownloadLinkRequest**(*session*, *path*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get a download link to a resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be downloaded
- **fields** – list of keys to be included in the response

Returns *LinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetTrashRequest**(*session*, *path=None*, *offset=0*, *limit=20*, *sort=None*, *preview_size=None*, *preview_crop=None*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get meta-information about a trash resource.

Parameters

- **path** – path to the trash resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response

Returns *TrashResourceObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**RestoreTrashRequest**(*session*, *path*, *dst_path=None*, *force_async=False*, *overwrite=False*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to restore trash.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the trash resource to be restored
- **dst_path** – destination path
- **force_async** – forces the operation to be executed asynchronously
- **overwrite** – *bool*, determines whether the destination can be overwritten
- **fields** – list of keys to be included in the response

Returns *LinkObject* or *OperationLinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**DeleteTrashRequest**(*session*, *path=None*, *force_async=False*, *fields=None*, ***kwargs*)

Bases: `yadisk.api.api_request.APIRequest`

A request to delete a trash resource.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **path** – path to the trash resource to be deleted
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response

Returns *OperationLinkObject* or *None*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**LastUploadedRequest**(*session*, *limit=20*, *media_type=None*, *preview_size=None*, *preview_crop=None*, *fields=None*, ***kwargs*)

Bases: `yadisk.api.api_request.APIRequest`

A request to get the list of latest uploaded files sorted by upload date.

Parameters

- **session** – an instance of `requests.Session` with prepared headers
- **limit** – maximum number of elements in the list
- **media_type** – type of files to include in the list
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response

Returns *LastUploadedResourceListObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**CopyRequest**(*session*, *src_path*, *dst_path*, *overwrite=False*, *force_async=False*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to copy a file or a directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **src_path** – source path
- **dst_path** – destination path
- **overwrite** – if *True* the destination path can be overwritten, otherwise, an error will be raised
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response

Returns *LinkObject* or *OperationLinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetMetaRequest**(*session*, *path*, *limit=None*, *offset=None*, *preview_size=None*, *preview_crop=None*, *sort=None*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get meta-information about a resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **limit** – number of children resources to be included in the response
- **offset** – number of children resources to be skipped in the response
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **sort** – *str*, field to be used as a key to sort children resources
- **fields** – list of keys to be included in the response

Returns *ResourceObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetUploadLinkRequest**(*session*, *path*, *overwrite=False*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to get an upload link.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to be uploaded at
- **overwrite** – *bool*, determines whether to overwrite the destination
- **fields** – list of keys to be included in the response

Returns *ResourceUploadLinkObject*

process_json(*js*)

Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**MkdirRequest**(*session*, *path*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to create a new directory.

Parameters

- **path** – path to the directory to be created
- **fields** – list of keys to be included in the response

Returns *LinkObject*

process_json(*js*)

Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**PublishRequest**(*session*, *path*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to make a resource public.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be published
- **fields** – list of keys to be included in the response

Returns *LinkObject*

process_json(*js*)

Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**UploadURLRequest**(*session*, *url*, *path*, *disable_redirects=False*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to upload a file from URL.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **url** – source URL
- **path** – destination path
- **disable_redirects** – *bool*, forbid redirects
- **fields** – list of keys to be included in the response

Returns *OperationLinkObject*

process_json(*js*)

Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**DeleteRequest**(*session*, *path*, *permanently=False*, *md5=None*, *force_async=False*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to delete a file or a directory.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource to be removed
- **permanently** – if *True*, the resource will be removed permanently, otherwise, it will be just moved to the trash
- **force_async** – forces the operation to be executed asynchronously
- **md5** – *str*, MD5 hash of the file to remove
- **fields** – list of keys to be included in the response

Returns *OperationLinkObject* or *None*

process_json(*js*)

Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**SaveToDiskRequest**(*session*, *public_key*, *name=None*, *path=None*, *save_path=None*, *force_async=False*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to save a public resource to the disk.

Parameters

- **session** – an instance of requests.Session with prepared headers

- **public_key** – public key or public URL of the resource
- **name** – filename of the saved resource
- **path** – path to the copied resource in the public folder
- **save_path** – path to the destination directory (downloads directory by default)
- **force_async** – forces the operation to be executed asynchronously
- **fields** – list of keys to be included in the response

Returns *LinkObject* or *OperationLinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetPublicMetaRequest**(*session*, *public_key*, *offset=0*, *limit=20*, *path=None*, *sort=None*, *preview_size=None*, *preview_crop=None*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to get meta-information about a public resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **public_key** – public key or public URL of the resource
- **path** – relative path to a resource in a public folder. By specifying the key of the published folder in *public_key*, you can request metainformation for any resource in the folder.
- **offset** – offset from the beginning of the list of nested resources
- **limit** – maximum number of nested elements to be included in the list
- **sort** – *str*, field to be used as a key to sort children resources
- **preview_size** – file preview size
- **preview_crop** – *bool*, allow preview crop
- **fields** – list of keys to be included in the response

Returns *PublicResourceObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**GetPublicDownloadLinkRequest**(*session*, *public_key*, *path=None*, *fields=None*, ***kwargs*)

Bases: yadisk.api.api_request.APIRequest

A request to get a download link for a public resource.

Parameters

- **session** – an instance of requests.Session with prepared headers

- **public_key** – public key or public URL of the resource
- **path** – relative path to the resource within the public folder
- **fields** – list of keys to be included in the response

Returns *LinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**MoveRequest**(*session*, *src_path*, *dst_path*, *force_async=False*, *overwrite=False*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to move a resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **src_path** – source path to be moved
- **dst_path** – destination path
- **force_async** – forces the operation to be executed asynchronously
- **overwrite** – *bool*, determines whether to overwrite the destination
- **fields** – list of keys to be included in the response

Returns *OperationLinkObject* or *LinkObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**FilesRequest**(*session*, *offset=0*, *limit=20*, *media_type=None*, *preview_size=None*, *preview_crop=None*, *sort=None*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get a flat list of all files (that doesn't include directories).

Parameters

- **session** – an instance of requests.Session with prepared headers
- **offset** – offset from the beginning of the list
- **limit** – number of list elements to be included
- **media_type** – type of files to include in the list
- **sort** – *str*, field to be used as a key to sort children resources
- **preview_size** – size of the file preview
- **preview_crop** – *bool*, cut the preview to the size specified in the *preview_size*
- **fields** – list of keys to be included in the response

Returns *FilesResourceListObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.resources.**PatchRequest**(*session*, *path*, *properties*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to update custom properties of a resource.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **path** – path to the resource
- **properties** – *dict*, custom properties to update
- **fields** – list of keys to be included in the response

Returns *ResourceObject*

prepare(**args*, ***kwargs*)
Prepare the request

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

class yadisk.api.operations.**GetOperationStatusRequest**(*session*, *operation_id*, *fields=None*, ***kwargs*)
Bases: yadisk.api.api_request.APIRequest

A request to get operation status.

Parameters

- **session** – an instance of requests.Session with prepared headers
- **operation_id** – operation ID or link
- **fields** – list of keys to be included in the response

Returns *OperationStatusObject*

process_json(*js*)
Process the JSON response.

Parameters **js** – *dict*, JSON response

Returns processed response, can be anything

CHAPTER 3

Changelog

- **Release 1.2.14 (2019-03-26)**
  - Fixed a *TypeError* in *get_public_** functions when passing *path* parameter (see issue #7)
  - Added *unlimited_autoupload_enabled* attribute for *DiskInfoObject*
- **Release 1.2.13 (2019-02-23)**
  - Added *md5* parameter for *remove()*
  - Added *UserPublicInfoObject*
  - Added *country* attribute for *UserObject*
  - Added *photoslice_time* attribute for *ResourceObject*, *PublicResourceObject* and *TrashResourceObject*
- **Release 1.2.12 (2018-10-11)**
  - Fixed *fields* parameter not working properly in *listdir()* (issue #4)
- **Release 1.2.11 (2018-06-30)**
  - Added the missing parameter *sort* for *get_meta()*
  - Added *file* and *antivirus_status* attributes for *ResourceObject*, *PublicResourceObject* and *TrashResourceObject*
  - Added *headers* parameter
  - Fixed a typo in *download()* and *download_public()* (issue #2)
  - Removed **args* parameter everywhere
- **Release 1.2.10 (2018-06-14)**
  - Fixed *timeout=None* behavior. *None* is supposed to mean 'no timeout' but in the older versions it was synonymous with the default timeout.
- **Release 1.2.9 (2018-04-28)**
  - Changed the license to LGPLv3 (see *COPYING* and *COPYING.lesser*)

  - Other package info updates

- **Release 1.2.8 (2018-04-17)**
  - Fixed a couple of typos: *PublicResourceListObject.items* and *TrashResourceListObject.items* had wrong types
  - Substitute field aliases in *fields* parameter when performing API requests (e.g. *embedded* -> *_embedded*)
- **Release 1.2.7 (2018-04-15)**
  - Fixed a file rewinding bug when uploading/downloading files after a retry
- **Release 1.2.6 (2018-04-13)**
  - Now caching *requests* sessions so that open connections can be reused (which can significantly speed things up sometimes)
  - Disable *keep-alive* when uploading/downloading files by default
- **Release 1.2.5 (2018-03-31)**
  - Fixed an off-by-one bug in *utils.auto_retry()* (which could sometimes result in *AttributeError*)
  - Retry the whole request for *upload()*, *download()* and *download_public()*
  - Set *stream=True* for *download()* and *download_public()*
  - Other minor fixes
- **Release 1.2.4 (2018-02-19)**
  - Fixed *TokenObject* having *exprires_in* instead of *expires_in* (fixed a typo)
- **Release 1.2.3 (2018-01-20)**
  - Fixed a *TypeError* when *WrongResourceTypeError* is raised
- **Release 1.2.2 (2018-01-19)**
  - *refresh_token()* no longer requires a valid or empty token.
- **Release 1.2.1 (2018-01-14)**
  - Fixed auto retries not working. Whoops.
- **Release 1.2.0 (2018-01-14)**
  - Fixed passing *n_retries=0* to *upload()*, *download()* and *download_public()*
  - *upload()*, *download()* and *download_public()* no longer return anything (see the docs)
  - Added *utils* module (see the docs)
  - Added *RetriableYaDiskError*, *WrongResourceTypeError*, *BadGatewayError* and *GatewayTimeoutError*
  - *listdir()* now raises *WrongResourceTypeError* instead of *NotADirectoryError*
- **Release 1.1.1 (2017-12-29)**
  - Fixed argument handling in *upload()*, *download()* and *download_public()*. Previously, passing *n_retries* and *retry_interval* would raise an exception (*TypeError*).
- **Release 1.1.0 (2017-12-27)**
  - Better exceptions (see the docs)
  - Added support for *force_async* parameter
  - Minor bug fixes

- **Release 1.0.8 (2017-11-29)**
  - Fixed yet another *listdir()* bug
- **Release 1.0.7 (2017-11-04)**
  - Added *install_requires* argument to *setup.py*
- **Release 1.0.6 (2017-11-04)**
  - Return *OperationLinkObject* in some functions
- **Release 1.0.5 (2017-10-29)**
  - Fixed *setup.py* to exclude tests
- **Release 1.0.4 (2017-10-23)**
  - Fixed bugs in *upload*, *download* and *listdir* functions
  - Set default *listdir limit* to *10000*
- **Release 1.0.3 (2017-10-22)**
  - Added settings
- **Release 1.0.2 (2017-10-19)**
  - Fixed *get_code_url* function (added missing parameters)
- **Release 1.0.1 (2017-10-18)**
  - Fixed a major bug in *GetTokenRequest* (added missing parameter)
- **Release 1.0.0 (2017-10-18)**
  - Initial release

CHAPTER 4

Donations

If you find this library helpful, please consider donating. Your donation will help the development of this library.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

y

yadisk.api, 47
yadisk.api.auth, 47
yadisk.api.disk, 48
yadisk.api.operations, 57
yadisk.api.resources, 49
yadisk.exceptions, 20
yadisk.functions.auth, 31
yadisk.functions.disk, 33
yadisk.functions.operations, 46
yadisk.functions.resources, 33
yadisk.objects, 23
yadisk.objects.auth, 24
yadisk.objects.disk, 24
yadisk.objects.operations, 30
yadisk.objects.resources, 25
yadisk.utils, 31

Index

A

APIRequest (*class in yadisk.api*), 47
auto_retry() (*in module yadisk.utils*), 31

B

BadGatewayError, 22
BadRequestError, 21

C

check_token() (*in module yadisk.functions.auth*), 31
check_token() (*yadisk.YaDisk method*), 5
clear_session_cache() (*yadisk.YaDisk method*), 5
CommentIDsObject (*class in yadisk.objects.resources*), 25
ConflictError, 21
copy() (*in module yadisk.functions.resources*), 33
copy() (*yadisk.YaDisk method*), 5
CopyRequest (*class in yadisk.api.resources*), 52

D

DeleteRequest (*class in yadisk.api.resources*), 54
DeleteTrashRequest (*class in yadisk.api.resources*), 51
DirectoryExistsError, 22
DiskInfoObject (*class in yadisk.objects.disk*), 24
DiskInfoRequest (*class in yadisk.api.disk*), 48
download() (*in module yadisk.functions.resources*), 34
download() (*yadisk.YaDisk method*), 6
download_public() (*in module yadisk.functions.resources*), 46
download_public() (*yadisk.YaDisk method*), 6

E

ErrorObject (*class in yadisk.objects*), 23
EXIFObject (*class in yadisk.objects.resources*), 25
exists() (*in module yadisk.functions.resources*), 34
exists() (*yadisk.YaDisk method*), 6

F

FieldValidationError, 22
FilesRequest (*class in yadisk.api.resources*), 56
FilesResourceListObject (*class in yadisk.objects.resources*), 25
ForbiddenError, 21

G

GatewayTimeoutError, 22
get_auth_url() (*in module yadisk.functions.auth*), 31
get_auth_url() (*yadisk.YaDisk method*), 7
get_code_url() (*in module yadisk.functions.auth*), 32
get_code_url() (*yadisk.YaDisk method*), 7
get_disk_info() (*in module yadisk.functions.disk*), 33
get_disk_info() (*yadisk.YaDisk method*), 7
get_download_link() (*in module yadisk.functions.resources*), 34
get_download_link() (*yadisk.YaDisk method*), 8
get_exception() (*in module yadisk.utils*), 31
get_files() (*in module yadisk.functions.resources*), 44
get_files() (*yadisk.YaDisk method*), 8
get_last_uploaded() (*in module yadisk.functions.resources*), 45
get_last_uploaded() (*yadisk.YaDisk method*), 8
get_meta() (*in module yadisk.functions.resources*), 34
get_meta() (*yadisk.YaDisk method*), 9
get_operation_status() (*in module yadisk.functions.operations*), 46
get_operation_status() (*yadisk.YaDisk method*), 9
get_public_download_link() (*in module yadisk.functions.resources*), 45
get_public_download_link() (*yadisk.YaDisk method*), 9
get_public_meta() (*in module*

yadisk.functions.resources), 40
get_public_meta() (*yadisk.YaDisk method*), 10
get_public_resources() (*in module yadisk.functions.resources*), 44
get_public_resources() (*yadisk.YaDisk method*), 10
get_public_type() (*in module yadisk.functions.resources*), 41
get_public_type() (*yadisk.YaDisk method*), 11
get_session() (*yadisk.YaDisk method*), 11
get_token() (*in module yadisk.functions.auth*), 32
get_token() (*yadisk.YaDisk method*), 11
get_trash_meta() (*in module yadisk.functions.resources*), 37
get_trash_meta() (*yadisk.YaDisk method*), 11
get_trash_type() (*in module yadisk.functions.resources*), 43
get_trash_type() (*yadisk.YaDisk method*), 12
get_type() (*in module yadisk.functions.resources*), 35
get_type() (*yadisk.YaDisk method*), 12
get_upload_link() (*in module yadisk.functions.resources*), 35
get_upload_link() (*yadisk.YaDisk method*), 12
GetDownloadLinkRequest (*class in yadisk.api.resources*), 50
GetMetaRequest (*class in yadisk.api.resources*), 52
GetOperationStatusRequest (*class in yadisk.api.operations*), 57
GetPublicDownloadLinkRequest (*class in yadisk.api.resources*), 55
GetPublicMetaRequest (*class in yadisk.api.resources*), 55
GetPublicResourcesRequest (*class in yadisk.api.resources*), 49
GetTokenRequest (*class in yadisk.api.auth*), 48
GetTrashRequest (*class in yadisk.api.resources*), 50
GetUploadLinkRequest (*class in yadisk.api.resources*), 53

I

import_fields() (*yadisk.objects.YaDiskObject method*), 23
InsufficientStorageError, 22
InternalServerError, 22
is_dir() (*in module yadisk.functions.resources*), 35
is_dir() (*yadisk.YaDisk method*), 12
is_file() (*in module yadisk.functions.resources*), 36
is_file() (*yadisk.YaDisk method*), 13
is_public_dir() (*in module yadisk.functions.resources*), 42
is_public_dir() (*yadisk.YaDisk method*), 13
is_public_file() (*in module yadisk.functions.resources*), 42
is_public_file() (*yadisk.YaDisk method*), 13
is_trash_dir() (*in module yadisk.functions.resources*), 43
is_trash_dir() (*yadisk.YaDisk method*), 13
is_trash_file() (*in module yadisk.functions.resources*), 43
is_trash_file() (*yadisk.YaDisk method*), 14

L

LastUploadedRequest (*class in yadisk.api.resources*), 51
LastUploadedResourceListObject (*class in yadisk.objects.resources*), 26
LinkObject (*class in yadisk.objects.resources*), 26
listdir() (*in module yadisk.functions.resources*), 36
listdir() (*yadisk.YaDisk method*), 14
LockedError, 21

M

make_session() (*yadisk.YaDisk method*), 14
MD5DifferError, 23
mkdir() (*in module yadisk.functions.resources*), 36
mkdir() (*yadisk.YaDisk method*), 14
MkdirRequest (*class in yadisk.api.resources*), 53
move() (*in module yadisk.functions.resources*), 39
move() (*yadisk.YaDisk method*), 15
MoveRequest (*class in yadisk.api.resources*), 56

N

NotAcceptableError, 21
NotFoundError, 21

O

OperationLinkObject (*class in yadisk.objects.resources*), 26
OperationStatusObject (*class in yadisk.objects.operations*), 30

P

ParentNotFoundError, 22
patch() (*in module yadisk.functions.resources*), 44
patch() (*yadisk.YaDisk method*), 15
PatchRequest (*class in yadisk.api.resources*), 57
PathExistsError, 22
PathNotFoundError, 22
prepare() (*yadisk.api.APIRequest method*), 47
prepare() (*yadisk.api.resources.PatchRequest method*), 57
process() (*yadisk.api.APIRequest method*), 47
process_json() (*yadisk.api.APIRequest method*), 47
process_json() (*yadisk.api.auth.GetTokenRequest method*), 48
process_json() (*yadisk.api.auth.RefreshTokenRequest method*), 48

process_json() (*yadisk.api.auth.RevokeTokenRequest method*), 48
process_json() (*yadisk.api.disk.DiskInfoRequest method*), 49
process_json() (*yadisk.api.operations.GetOperationStatusRequest method*), 57
process_json() (*yadisk.api.resources.CopyRequest method*), 52
process_json() (*yadisk.api.resources.DeleteRequest method*), 54
process_json() (*yadisk.api.resources.DeleteTrashRequest method*), 51
process_json() (*yadisk.api.resources.FilesRequest method*), 56
process_json() (*yadisk.api.resources.GetDownloadLinkRequest method*), 50
process_json() (*yadisk.api.resources.GetMetaRequest method*), 52
process_json() (*yadisk.api.resources.GetPublicDownloadLinkRequest method*), 56
process_json() (*yadisk.api.resources.GetPublicMetaRequest method*), 55
process_json() (*yadisk.api.resources.GetPublicResourcesRequest method*), 49
process_json() (*yadisk.api.resources.GetTrashRequest method*), 50
process_json() (*yadisk.api.resources.GetUploadLinkRequest method*), 53
process_json() (*yadisk.api.resources.LastUploadedRequest method*), 52
process_json() (*yadisk.api.resources.MkdirRequest method*), 53
process_json() (*yadisk.api.resources.MoveRequest method*), 56
process_json() (*yadisk.api.resources.PatchRequest method*), 57
process_json() (*yadisk.api.resources.PublishRequest method*), 53
process_json() (*yadisk.api.resources.RestoreTrashRequest method*), 51
process_json() (*yadisk.api.resources.SaveToDiskRequest method*), 55
process_json() (*yadisk.api.resources.UnpublishRequest method*), 49
process_json() (*yadisk.api.resources.UploadURLRequest method*), 54
public_exists() (*in module yadisk.functions.resources*), 41
public_exists() (*yadisk.YaDisk method*), 15
public_listdir() (*in module yadisk.functions.resources*), 41
public_listdir() (*yadisk.YaDisk method*), 16
PublicResourceListObject (*class in yadisk.objects.resources*), 29
PublicResourceObject (*class in yadisk.objects.resources*), 28
PublicResourcesListObject (*class in yadisk.objects.resources*), 26
publish() (*in module yadisk.functions.resources*), 39
publish() (*yadisk.YaDisk method*), 16
PublishRequest (*class in yadisk.api.resources*), 53

R

refresh_token() (*in module yadisk.functions.auth*), 32
refresh_token() (*yadisk.YaDisk method*), 16
RefreshTokenRequest (*class in yadisk.api.auth*), 47
remove() (*in module yadisk.functions.resources*), 37
remove() (*yadisk.YaDisk method*), 16
remove_alias() (*yadisk.objects.YaDiskObject method*), 23
remove_field() (*yadisk.objects.YaDiskObject method*), 23
remove_trash() (*in module yadisk.functions.resources*), 39
remove_trash() (*yadisk.YaDisk method*), 17
ResourceIsLockedError, 22
ResourceListObject (*class in yadisk.objects.resources*), 27
ResourceObject (*class in yadisk.objects.resources*), 27
ResourceUploadLinkObject (*class in yadisk.objects.resources*), 28
restore_trash() (*in module yadisk.functions.resources*), 38
restore_trash() (*yadisk.YaDisk method*), 17
RestoreTrashRequest (*class in yadisk.api.resources*), 50
RetriableYaDiskError, 21
revoke_token() (*in module yadisk.functions.auth*), 33
revoke_token() (*yadisk.YaDisk method*), 18
RevokeTokenRequest (*class in yadisk.api.auth*), 48

S

save_to_disk() (*in module yadisk.functions.resources*), 40
save_to_disk() (*yadisk.YaDisk method*), 18
SaveToDiskRequest (*class in yadisk.api.resources*), 54
send() (*yadisk.api.APIRequest method*), 47
set_alias() (*yadisk.objects.YaDiskObject method*), 23
set_field_type() (*yadisk.objects.YaDiskObject method*), 23
set_field_types() (*yadisk.objects.YaDiskObject method*), 23

ShareInfoObject (*class in yadisk.objects.resources*), 28
SystemFoldersObject (*class in yadisk.objects.disk*), 24

T

TokenObject (*class in yadisk.objects.auth*), 24
TokenRevokeStatusObject (*class in yadisk.objects.auth*), 24
TooManyRequestsError, 21
trash_exists() (*in module yadisk.functions.resources*), 38
trash_exists() (*yadisk.YaDisk method*), 18
trash_listdir() (*in module yadisk.functions.resources*), 42
trash_listdir() (*yadisk.YaDisk method*), 18
TrashResourceListObject (*class in yadisk.objects.resources*), 30
TrashResourceObject (*class in yadisk.objects.resources*), 29

U

UnauthorizedError, 21
UnavailableError, 22
UnknownYaDiskError, 21
unpublish() (*in module yadisk.functions.resources*), 40
unpublish() (*yadisk.YaDisk method*), 19
UnpublishRequest (*class in yadisk.api.resources*), 49
UnsupportedMediaError, 21
upload() (*in module yadisk.functions.resources*), 37
upload() (*yadisk.YaDisk method*), 19
upload_url() (*in module yadisk.functions.resources*), 45
upload_url() (*yadisk.YaDisk method*), 19
UploadURLRequest (*class in yadisk.api.resources*), 54
UserObject (*class in yadisk.objects.disk*), 25
UserPublicInfoObject (*class in yadisk.objects.disk*), 25

W

WrongResourceTypeError, 21

Y

YaDisk (*class in yadisk*), 5
yadisk.api (*module*), 47
yadisk.api.auth (*module*), 47
yadisk.api.disk (*module*), 48
yadisk.api.operations (*module*), 57
yadisk.api.resources (*module*), 49
yadisk.exceptions (*module*), 20
yadisk.functions.auth (*module*), 31
yadisk.functions.disk (*module*), 33
yadisk.functions.operations (*module*), 46
yadisk.functions.resources (*module*), 33
yadisk.objects (*module*), 23
yadisk.objects.auth (*module*), 24
yadisk.objects.disk (*module*), 24
yadisk.objects.operations (*module*), 30
yadisk.objects.resources (*module*), 25
yadisk.utils (*module*), 31
YaDiskError, 20
YaDiskObject (*class in yadisk.objects*), 23