YaDisk Documentation

Выпуск 1.3.0

Ivan Konovalov

июл. 06, 2019

Содежание:

Глава 1

Введение

YaDisk-async - это модифицированная версия YaDisk с поддержкой `async/await`, использует *aiohttp* вместо *requests*.

Использование библиотеки почти не отличается, не считая того, что закрывать все сессии нужно вручную (может быть сделано с помощью *YaDisk.close* или через конструкцию `async with`).

1.1 Установка

```
pip install yadisk-async
```

или

```
python setup.py install
```

1.2 Примеры

```
import yadisk_async

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

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

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

# Print files and directories at "/some/path"
```

(continues on next page)

(продолжение с предыдущей страницы)

```
print([i async for i in await y.listdir("/some/path")])

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

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

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

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

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

# Always remember to close all the connections or you'll get a warning
await y.close()
```

1.2.1 Получение токена через код подтверждения

```
import asyncio
import sys
import yadisk_async

async def main():
    async with yadisk_async.YaDisk("application-id>", "<application-secret>") as y:
        url = y.get_code_url()

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

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

        y.token = response.access_token

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

loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

1.2.2 Рекурсивная загрузка файлов

```
import asyncio
import posixpath
import os
import yadisk_async

def recursive_upload(from_dir, to_dir, n_parallel_requests=5):
    loop = asyncio.get_event_loop()

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

    try:
        async def upload_files(queue):
            while queue:
                in_path, out_path = queue.pop(0)

                print("Uploading %s -> %s" % (in_path, out_path))

                try:
                    await y.upload(in_path, out_path)
                except yadisk_async.exceptions.PathExistsError:
                    print("%s already exists" % (out_path,))

        async def create_dirs(queue):
            while queue:
                path = queue.pop(0)

                print("Creating directory %s" % (path,))

                try:
                    await y.mkdir(path)
                except yadisk_async.exceptions.PathExistsError:
                    print("%s already exists" % (path,))

        mkdir_queue = []
        upload_queue = []

        print("Creating directory %s" % (to_dir,))

        try:
            loop.run_until_complete(y.mkdir(to_dir))
        except yadisk_async.exceptions.PathExistsError:
            print("%s already exists" % (to_dir,))

        for root, dirs, files in os.walk(from_dir):
            rel_dir_path = root.split(from_dir)[1].strip(os.path.sep)
            rel_dir_path = rel_dir_path.replace(os.path.sep, "/")
            dir_path = posixpath.join(to_dir, rel_dir_path)

            for dirname in dirs:
                mkdir_queue.append(posixpath.join(dir_path, dirname))

            for filename in files:
                out_path = posixpath.join(dir_path, filename)
                rel_dir_path_sys = rel_dir_path.replace("/", os.path.sep)
                in_path = os.path.join(from_dir, rel_dir_path_sys, filename)
```

(continues on next page)

(продолжение с предыдущей страницы)

```
                upload_queue.append((in_path, out_path))

            tasks = [upload_files(upload_queue) for i in range(n_parallel_requests)]
            tasks.extend(create_dirs(mkdir_queue) for i in range(n_parallel_requests))

            loop.run_until_complete(asyncio.gather(*tasks))
    finally:
        loop.run_until_complete(y.close())

from_dir = input("Directory to upload: ")
to_dir = input("Destination directory: ")

recursive_upload(from_dir, to_dir, 5)
```

1.2.3 Задание пользовательских свойств файлов

```
import asyncio
import yadisk_async

async def main():
    async with yadisk_async.YaDisk(token="<application-token>") as y:
        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 = await 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}
            await y.patch(path, properties)
            print("Everything's back as usual")

loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

1.2.4 Очищение корзины

```
import asyncio
import sys
import yadisk_async

async def main():
    async with yadisk_async.YaDisk(token="<application-token>") as y:
```

(continues on next page)

(продолжение с предыдущей страницы)

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

        if answer.lower() in ("y", "yes"):
            print("Emptying the trash bin...")
            operation = await 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 = await y.get_operation_status(operation.href)

                if status == "in-progress":
                    await asyncio.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")

loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

Глава 2

Документация

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

Реализует доступ к REST API Яндекс.Диска.

Примечание: Не забывайте вызывать *YaDisk.close* или используйте *async with*, чтобы закрыть все соединения. Иначе, вы можете получить предупреждение.

В оригинальной библиотеке это делалось в деструкторе, но т.к. aiohttp.ClientSession.close - сопрограмма, здесь этого сделать нельзя, поэтому приходится делать это явно.

Параметры

- id – идентификатор приложения
- secret – пароль приложения
- token – токен

Переменные

- id – *str*, идентификатор приложения
- secret – *str*, пароль приложения
- token – *str*, токен

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

Проверяет, действителен ли токен.

Параметры

- token – токен, подлежащий проверке, то же самое, что *self.token* при *None*
- timeout – *float* или *tuple*, таймаут запроса
- headers – *dict* или *None*, дополнительные заголовки запроса
- n_retries – *int*, максимальное число повторных попыток запроса

- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`clear_session_cache()`
Очищает кэш сессий. Неиспользуемые сессии НЕ будут закрыты.

Этот метод не является сопрограммой.

`close()`
Закрывет все сессии и очищает кэш сессий. Не вызывайте этот метод, пока другие потоки используют этот объект.

Этот метод неявно вызывается конструкцией *async with*.

`copy(`*src_path*, *dst_path*, ***kwargs*`)`
Копирует *src_path* в *dst_path*. Если операция выполняется асинхронно, возвращает ссылку на операцию, иначе, возвращает ссылку на новый ресурс.

Параметры

- `src_path` – исходный путь
- `dst_path` – путь назначения
- `overwrite` – если *True*, путь назначения может быть перезаписан, иначе будет вызвана ошибка
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

`download(`*src_path*, *path_or_file*, ***kwargs*`)`
Скачивает файл.

Параметры

- `src_path` – исходный путь
- `path_or_file` – путь назначения или файл-подобный объект
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`download_public(`*public_key*, *file_or_path*, ***kwargs*`)`
Скачивает публичный ресурс.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `file_or_path` – путь назначения или файл-подобный объект
- `path` – относительный путь к ресурсу внутри публичной папки

- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`exists(`*path*, ***kwargs*`)`

Проверяет, существует ли *path*.

Параметры

- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`get_auth_url(`***kwargs*`)`

Получает URL для аутентификации для пользователя.

Этот метод не является сопрограммой.

Параметры

- `type` – тип ответа («code», чтобы получить код подтверждения или «token», чтобы получить токен автоматически)
- `device_id` – уникальный идентификатор устройства, от 6 до 50 символов
- `device_name` – имя устройства, не более 100 символов
- `display` – указывает использовать облегчённую вёрстку, обрабатывает только «popup», остальные значения игнорируются
- `login_hint` – username или email аккаунта, для которого будет получен токен
- `scope` – список разрешений для приложения
- `optional_scope` – список опциональных разрешений для приложения
- `force_confirm` – Если *True*, пользователь должен будет разрешить доступ к аккаунту, даже если он уже это сделал до этого
- `state` – Строка состояния, которую Яндекс.OAuth возвращает без изменений (<= 1024 символов)

Результат URL для аутентификации

`get_code_url(`***kwargs*`)`

Получает URL для получения пользователем кода подтверждения. Он может быть использован для получения токена.

Этот метод не является сопрограммой.

Параметры

- `device_id` – уникальный идентификатор устройства, от 6 до 50 символов
- `device_name` – имя устройства, не более 100 символов

- `display` – указывает использовать облегчённую вёрстку, обрабатывает только «popup», остальные значения игнорируются
- `login_hint` – username или email аккаунта, для которого будет получен токен
- `scope` – список разрешений для приложения
- `optional_scope` – список опциональных разрешений для приложения
- `force_confirm` – Если *True*, пользователь должен будет разрешить доступ к аккаунту, даже если он уже это сделал до этого
- `state` – Строка состояния, которую Яндекс.OAuth возвращает без изменений (<= 1024 символов)

Результат URL для аутентификации

`get_disk_info(**kwargs)`
Получает информацию о диске.

Параметры

- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *DiskInfoObject*

`get_download_link(path, **kwargs)`
Получает ссылку на скачивание файла (или папки).

Параметры

- `path` – путь к ресурсу
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

`get_files(**kwargs)`
Получить плоский список всех файлов (без папок).

Параметры

- `offset` – отступ от начала списка
- `limit` – максимальное количество элементов списка
- `media_type` – тип файлов, которые будут включены в список
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла

- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *ResourceObject*

`get_last_uploaded(**kwargs)`

Получает список последних загруженных файлов, отсортированный по дате загрузки.

Параметры

- `limit` – максимальное число элементов в списке
- `media_type` – тип файлов, которые будут включены в список
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *LastUploadedResourceListObject*

`get_meta(path, **kwargs)`

Получает мета-информацию о ресурсе.

Параметры

- `path` – путь к ресурсу
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *ResourceObject*

get_operation_status(*operation_id*, ***kwargs*)
Получает статус операции.

Параметры

- `operation_id` – идентификатор операции или ссылка на нее
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

get_public_download_link(*public_key*, ***kwargs*)
Получает ссылку на скачивание публичного ресурса.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

get_public_meta(*public_key*, ***kwargs*)
Получает мета-информацию о публичном ресурсе.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки. Указывая ключ опубликованной папки через *public_key*, вы можете запросить метаинформацию любого ресурса внутри неё.
- `offset` – отступ от начала списка вложенных ресурсов
- `limit` – максимальное количество элементов списка вложенных ресурсов
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, разрешить обрезку превью
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *PublicResourceObject*

`get_public_resources`(***kwargs*)
Получает список публичных ресурсов.

Параметры

- `offset` – отступ от начала списка
- `limit` – максимальное число элементов в списке
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `type` – фильтр по типу ресурса («file» или «dir»)
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *PublicResourcesListObject*

`get_public_type`(*public_key*, ***kwargs*)
Получает тип публичного ресурса.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

`get_session`(*token=None*)
То же, что и *YaDisk.make_session*, использует кэш.

Этот метод не является сопрограммой.

Результат *yadisk_async.session.SessionWithHeaders*, отдельные объекты для разных потоков

`get_token`(*code*, ***kwargs*)
Получает новый токен.

Параметры

- `code` – код подтверждения
- `device_id` – уникальный идентификатор устройства (между 6 и 50 символами)
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса

- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TokenObject*

`get_trash_meta`(*path*, ***kwargs*)
Получает мета-информацию о ресурсе корзины.

Параметры

- `path` – путь к ресурсу корзины
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TrashResourceObject*

`get_trash_type`(*path*, ***kwargs*)
Получает тип ресурса корзины.

Параметры

- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

`get_type`(*path*, ***kwargs*)
Получает тип ресурса

Параметры

- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

`get_upload_link(path, **kwargs)`
Получает ссылку для загрузки файла на диск при помощи PUT запроса.

Параметры

- `path` – путь назначения
- `overwrite` – *bool*, определяет, перезаписывать путь назначения или нет
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

`is_dir(path, **kwargs)`
Проверяет, является ли *path* папкой.

Параметры

- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

`is_file(path, **kwargs)`
Проверяет, является ли *path* файлом.

Параметры

- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является файлом, *False*, в остальных случаях (даже если ресурс не существует)

`is_public_dir(public_key, **kwargs)`
Проверяет, является ли *public_key* публичной папкой.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса

- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *public_key* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

`is_public_file(`*public_key*, ***kwargs*`)`
Проверяет, является ли *public_key* публичным файлом.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *public_key* является файлом, *False*, в остальных случаях (даже если ресурс не существует)

`is_trash_dir(`*path*, ***kwargs*`)`
Проверяет, является ли *path* папкой в корзине.

Параметры

- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

`is_trash_file(`*path*, ***kwargs*`)`
Проверяет, является ли *path* файлом в корзине.

Параметры

- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

`listdir(`*path*, ***kwargs*`)`
Получает содержимое *path*.

Параметры

- `path` – путь к папке

- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *ResourceObject*

`make_session`(*token=None*)

Готовит объект *yadisk_async.session.SessionWithHeaders* с заголовками, необходимыми для API.

Этот метод не является сопрограммой.

Параметры `token` – токен, то же самое, что *self.token*, если *None*

Результат *yadisk_async.session.SessionWithHeaders*

`mkdir`(*path*, ***kwargs*)

Создаёт новую папку.

Параметры

- `path` – путь к папке, подлежащей созданию
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*

`move`(*src_path*, *dst_path*, ***kwargs*)

Перемещает *src_path* в *dst_path*.

Параметры

- `src_path` – исходный путь, подлежащий перемещению
- `dst_path` – путь назначения
- `overwrite` – *bool*, определяет, перезаписывать путь назначения или нет
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса

- `retry_interval` – задержка между повторными попытками в секундах

Результат *OperationLinkObject* или *LinkObject*

`patch`(*path*, *properties*, ***kwargs*)
Обновляет пользовательские свойства ресурса.

Параметры

- `path` – путь к ресурсу
- `properties` – *dict*, новые пользовательские свойства ресурса
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *ResourceObject*

`public_exists`(*public_key*, ***kwargs*)
Проверяет, существует ли публичный ресурс.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`public_listdir`(*public_key*, ***kwargs*)
Получает содержимое публичной папки.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу в публичной папке. Указывая ключ опубликованной папки через *public_key*, вы можете запросить содержимое любой вложенной папки.
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса

- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *PublicResourceObject*

`publish(path, **kwargs)`
Делает ресурс публичным.

Параметры

- `path` – путь к публикуемому ресурсу
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*, ссылка на ресурс

`refresh_token(refresh_token, **kwargs)`
Обновляет существующий токен.

Параметры

- `refresh_token` – refresh-токен, полученный вместе с токеном
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TokenObject*

`remove(path, **kwargs)`
Удаляет ресурс.

Параметры

- `path` – путь к удаляемому ресурсу
- `permanently` – если *True*, ресурс будет безвозвратно удалён, иначе ресурс будет перемещён в корзину
- `md5` – *str*, MD5 хэш файла, подлежащего удалению
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *OperationLinkObject*, если операция выполняется асинхронно, иначе *None*

remove_trash(*path*, ***kwargs*)
Удаляет ресурс корзины.

Параметры

- path – путь к ресурсу корзины, подлежащий удалению
- force_async – заставляет выполнять операцию асинхронно
- fields – список ключей, которые будут включены в ответ
- timeout – *float* или *tuple*, таймаут запроса
- headers – *dict* или *None*, дополнительные заголовки запроса
- n_retries – *int*, максимальное число повторных попыток запроса
- retry_interval – задержка между повторными попытками в секундах

Результат *OperationLinkObject*, если операция выполняется асинхронно, иначе *None*

restore_trash(*path*, *dst_path=None*, ***kwargs*)
Восстанавливает ресурс корзины. Возвращает ссылку на новый ресурс или ссылку на асинхронную операцию.

Параметры

- path – путь к ресурсу корзины, подлежащий удалению
- dst_path – путь назначения
- overwrite – *bool*, определяет может ли путь назначения быть перезаписан
- force_async – заставляет выполнять операцию асинхронно
- fields – список ключей, которые будут включены в ответ
- timeout – *float* или *tuple*, таймаут запроса
- headers – *dict* или *None*, дополнительные заголовки запроса
- n_retries – *int*, максимальное число повторных попыток запроса
- retry_interval – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

revoke_token(*token=None*, ***kwargs*)
Отзывает токен.

Параметры

- token – токен, подлежащий отзыву, то же самое, что *self.token*, если *None*
- timeout – *float* или *tuple*, таймаут запроса
- headers – *dict* или *None*, дополнительные заголовки запроса
- n_retries – *int*, максимальное число повторных попыток запроса
- retry_interval – задержка между повторными попытками в секундах

Результат *TokenRevokeStatusObject*

save_to_disk(*public_key*, ***kwargs*)
Сохраняет публичный ресурс на диск. Возвращает ссылку на операцию, если сохранение выполняется асинхронно, или возвращает ссылку на ресурс.

Параметры

- `public_key` – публичный ключ или URL к ресурсу
- `name` – имя файла/папки, под которым будет сохранён ресурс
- `path` – путь к копируемому ресурсу в публичной папке
- `save_path` – путь к папке назначения (загрузки по умолчанию)
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

`trash_exists(`*path*, ***kwargs*`)`
Проверяет, существует ли *path* в корзине.

Параметры

- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`trash_listdir(`*path*, ***kwargs*`)`
Получает содержимое папки в корзине.

Параметры

- `path` – путь к папке в корзине
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *TrashResourceObject*

`unpublish`(*path*, ***kwargs*)
Делает публичный ресурс приватным.

Параметры

- `path` – путь к ресурсу, подлежащему депубликации
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*, ссылка на ресурс

`upload`(*path_or_file*, *dst_path*, ***kwargs*)
Загружает файл на диск.

Параметры

- `path_or_file` – путь к файлу или файл-подобный объект для загрузки
- `dst_path` – путь назначения
- `overwrite` – если *True*, путь назначения может быть перезаписан, иначе будет вызвана ошибка
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`upload_url`(*url*, *path*, ***kwargs*)
Загружает файл на диск по URL.

Параметры

- `url` – исходный URL
- `path` – путь назначения
- `disable_redirects` – *bool*, запретить делать перенаправления
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *OperationLinkObject*, ссылка на асинхронную операцию

2.1 Общие параметры

Почти все методы *YaDisk* (те, которые принимают ***kwargs*) принимают некоторые дополнительные параметры:

- **n_retries** - *int*, максимальное число повторных попыток запроса
- **retry_delay** - *float*, задержка между повторными попытками (в секундах)
- **headers** - *dict* или *None*, дополнительные заголовки запроса

Параметры *aiohttp*, такие как *timeout*, *proxies* и подобные так же принимаются (см. `aiohttp.request()`).

Это так же применяется для низкоуровневых функций и объектов запросов API.

2.2 Настройки

Следующие настройки в модуле *yadisk_async.settings* могут быть получены и изменены:

- **DEFAULT_TIMEOUT** - `aiohttp.ClientTimeout`, default timeout for requests.
- **DEFAULT_N_RETRIES** - *int*, максимальное число повторных попыток запроса по умолчанию
- **DEFAULT_RETRY_DELAY** - *float*, стандартная задержка между повторными попытками
- **DEFAULT_UPLOAD_TIMEOUT** - аналогично *DEFAULT_TIMEOUT*, но для функции *upload*
- **DEFAULT_UPLOAD_RETRY_INTERVAL** - аналогично *DEFAULT_RETRY_INTERVAL*, но для функции *upload*

2.3 Исключения

Кроме исключений, перечисленных ниже, запросы к API могут также вызвать исключения *aiohttp*.

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

Базовые классы: `Exception`

Базовый класс для всех исключений в этой библиотеке.

Переменные

- `error_type` – *str*, уникальный код ошибки, полученный от API
- `response` – объект `aiohttp.ClientResponse`

Параметры

- `error_type` – *str*, уникальный код ошибки, полученный от API
- `msg` – *str*, сообщение исключения
- `response` – объект `aiohttp.ClientResponse`

exception yadisk_async.exceptions.RetriableYaDiskError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается в случае, если произошла ошибка, но имеет смысл повторить запрос.

exception yadisk_async.exceptions.UnknownYaDiskError(*msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.RetriableYaDiskError*

Вызывается, когда запрос не удался, но не содержит информации об ошибке.

exception yadisk_async.exceptions.WrongResourceTypeError(*msg=''*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда ожидался ресурс другого типа (например, файл вместо папки).

exception yadisk_async.exceptions.BadRequestError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 400.

exception yadisk_async.exceptions.UnauthorizedError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 401.

exception yadisk_async.exceptions.ForbiddenError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 403.

exception yadisk_async.exceptions.NotFoundError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 404.

exception yadisk_async.exceptions.NotAcceptableError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 406.

exception yadisk_async.exceptions.ConflictError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 409.

exception yadisk_async.exceptions.UnsupportedMediaError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 415.

exception yadisk_async.exceptions.LockedError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 423.

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

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 429.

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

Базовые классы: *yadisk_async.exceptions.RetriableYaDiskError*

Вызывается, когда сервер вернул код 500.

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

Базовые классы: *yadisk_async.exceptions.RetriableYaDiskError*

Вызывается, когда сервер вернул код 502.

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

Базовые классы: *yadisk_async.exceptions.RetriableYaDiskError*

Вызывается, когда сервер вернул код 503.

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

Базовые классы: *yadisk_async.exceptions.RetriableYaDiskError*

Вызывается, когда сервер вернул код 504.

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

Базовые классы: *yadisk_async.exceptions.YaDiskError*

Вызывается, когда сервер вернул код 509.

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

Базовые классы: *yadisk_async.exceptions.NotFoundError*

Вызывается, когда запрашиваемый путь не существует.

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

Базовые классы: *yadisk_async.exceptions.ConflictError*

Вызывается *mkdir*, *upload* и т.д. когда родительская папка не существует.

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

Базовые классы: *yadisk_async.exceptions.ConflictError*

Вызывается, когда запрашиваемый путь уже существует.

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

Базовые классы: *yadisk_async.exceptions.PathExistsError*

Вызывается, когда папка уже существует.

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

Базовые классы: *yadisk_async.exceptions.BadRequestError*

Вызывается, когда запрос содежит поля с некорректными данными.

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

Базовые классы: *yadisk_async.exceptions.LockedError*

Вызывается, когда запрашиваемый ресурс заблокирован другой операцией.

exception yadisk_async.exceptions.MD5DifferError(*error_type=None*, *msg=''*, *response=None*)

Базовые классы: *yadisk_async.exceptions.ConflictError*

Вызывается, когда MD5 хэш удаляемого ресурса не совпадает с указанным.

2.4 Объекты

class yadisk_async.objects.YaDiskObject(*field_types=None*)

Базовый класс для всех объектов, реализующий объекты, возвращаемые REST API Яндекс.Диска. У наследующего объекта фиксированное количество полей, каждое со своим типом. Поддерживает доступ по индексу и через точку.

Параметры `field_types` – *dict* или *None*

import_fields(*source_dict*)

Задаёт значения всех полей объекта из *source_dict*. Все остальные ключи игнорируются.

Параметры `source_dict` – *dict* или *None* (тогда ничего не будет сделано)

remove_alias(*alias*)

Удаляет псевдоним.

Параметры `alias` – *str*

remove_field(*field*)

Удаляет поле.

Параметры `field` – *str*

set_alias(*alias*, *name*)

Задаёт псевдоним.

Параметры

- `alias` – *str*, псевдоним
- `name` – *str*, имя поля

set_field_type(*field*, *type*)

Задаёт тип поля.

Параметры

- `field` – *str*
- `type` – тип данных или factory

set_field_types(*field_types*)

Задаёт типы полей объекта

Параметры `field_types` – *dict*, где ключи - это наименования полей, а значения - это типы

class yadisk_async.objects.ErrorObject(*error=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Реализует объект ошибки REST API Яндекс.Диска.

Параметры `error` – *dict* или *None*

Переменные

- `message` – *str*, человеко-читаемое сообщение ошибки
- `description` – *str*, техническое описание ошибки
- `error` – *str*, уникальный код ошибки

`class yadisk_async.objects.auth.TokenObject(`*token=None*`)`

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект токена.

Параметры `token` – *dict* или *None*

Переменные

- `access_token` – *str*, строка токена
- `refresh_token` – *str*, refresh-токен
- `token_type` – *str*, тип токена
- `expires_in` – *int*, количество времени, на которое выдаётся токен

`class yadisk_async.objects.auth.TokenRevokeStatusObject(`*token_revoke_status=None*`)`

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Результат запроса по отзыву токена.

Параметры `token_revoke_status` – *dict* или *None*

Переменные `status` – *str*, статус операции

`class yadisk_async.objects.disk.DiskInfoObject(`*disk_info=None*`)`

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект информации о диске.

Параметры `disk_info` – *dict* или *None*

Переменные

- `max_file_size` – *int*, максимальный поддерживаемый размер файла (в байтах)
- `unlimited_autoupload_enabled` – *bool*, признак включенной безлимитной автозагрузки с мобильных устройств
- `total_space` – *int*, общий размер диска (в байтах)
- `trash_size` – *int*, размер, занятый мусором (в байтах), часть *used_space*
- `is_paid` – *bool*, признак платного аккаунта
- `used_space` – *int*, количество занятого места (в байтах)
- `system_folders` – *SystemFoldersObject*, пути к системным папкам
- `user` – *UserObject*, владелец диска
- `revision` – *int*, текущая ревизия Яндекс.Диска

`class yadisk_async.objects.disk.SystemFoldersObject(`*system_folders=None*`)`

Базовые классы: yadisk_async.objects.yadisk_object.YaDiskObject

Объект, содержащий пути к системным папкам.

Параметры system_folders – *dict* или *None*

Переменные

- odnoklassniki – *str*, путь к папке Одноклассников
- google – *str*, путь к папке Google+
- instagram – *str*, путь к папке Instagram
- vkontakte – *str*, путь к папке ВКонтакте
- mailru – *str*, путь к папке Моего Мира
- facebook – *str*, путь к папке Facebook
- social – *str*, путь к папке социальных сетей
- screenshots – *str*, путь к папке скриншотов
- photostream – *str*, путь к папке фотокамеры

class yadisk_async.objects.disk.UserObject(*user=None*)

Базовые классы: yadisk_async.objects.yadisk_object.YaDiskObject

Объект пользователя.

Параметры user – *dict* или *None*

Переменные

- country – *str*, страна пользователя
- login – *str*, логин пользователя
- display_name – *str*, отображаемое имя пользователя
- uid – *str*, уникальный идентификатор пользователя

class yadisk_async.objects.disk.UserPublicInfoObject(*public_user_info=None*)

Базовые классы: *yadisk_async.objects.disk.UserObject*

Публичная информация о пользователе. Наследуется от *UserObject* для совместимости.

Параметры public_user_info – *dict* или *None*

Переменные

- login – *str*, логин пользователя
- display_name – *str*, отображаемое имя пользователя
- uid – *str*, уникальный идентификатор пользователя

class yadisk_async.objects.resources.CommentIDsObject(*comment_ids=None*)

Базовые классы: yadisk_async.objects.yadisk_object.YaDiskObject

Список идентификаторов комментариев.

Параметры comment_ids – *dict* или *None*

Переменные

- private_resource – *str*, идентификатор комментария для приватных ресурсов

- `public_resource` – *str*, идентификатор комментария для публичных ресурсов

class yadisk_async.objects.resources.EXIFObject(*exif=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект метаданных EXIF.

Параметры `exif` – *dict* или *None*

Переменные `date_time` – `datetime.datetime`, дата съёмки

class yadisk_async.objects.resources.FilesResourceListObject(*files_resource_list=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Плоский список файлов.

Параметры `files_resource_list` – *dict* или *None*

Переменные

- `items` – *list*, плоский список файлов (*ResourceObject*)
- `limit` – *int*, максимальное число элементов в списке
- `offset` – *int*, отступ от начала списка

class yadisk_async.objects.resources.LastUploadedResourceListObject(*last_uploaded_resources_list=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Список последних загруженных файлов.

Параметры `last_uploaded_resources_list` – *dict* или *None*

Переменные

- `items` – *list*, список ресурсов (*ResourceObject*)
- `limit` – *int*, максимальное число элементов в списке

class yadisk_async.objects.resources.LinkObject(*link=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект ссылки.

Параметры `link` – *dict* или *None*

Переменные

- `href` – *str*, URL ссылки
- `method` – *str*, HTTP метод
- `templated` – *bool*, признак шаблонизированного URL

class yadisk_async.objects.resources.OperationLinkObject(*link=None*)

Базовые классы: *yadisk_async.objects.resources.LinkObject*

Объект ссылки на операцию.

Параметры `link` – *dict* или *None*

Переменные

- `href` – *str*, URL ссылки
- `method` – *str*, HTTP метод

- `templated` – *bool*, признак шаблонизированного URL

class yadisk_async.objects.resources.PublicResourcesListObject(*public_resources_list=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Список публичных ресурсов.

Параметры `public_resources_list` – *dict* или *None*

Переменные

- `items` – *list*, список публичных ресурсов (*PublicResourceObject*)
- `type` – *str*, тип ресурса по которому фильтровать
- `limit` – *int*, максимальное число элементов в списке
- `offset` – *int*, отступ от начала списка

class yadisk_async.objects.resources.ResourceListObject(*resource_list=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Список ресурсов.

Параметры `resource_list` – *dict* или *None*

Переменные

- `sort` – *str*, тип сортировки
- `items` – *list*, список ресурсов (*ResourceObject*)
- `limit` – *int*, максимальное число элементов в списке
- `offset` – *int*, отступ от начала списка
- `path` – *str*, путь к папке, содержащей элементы списка
- `total` – *int*, количество элементов списка

class yadisk_async.objects.resources.ResourceObject(*resource=None*)

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект ресурса.

Параметры `resource` – *dict* или *None*

Переменные

- `antivirus_status` – *str*, статус проверки антивирусом
- `file` – *str*, URL для скачивания файла
- `size` – *int*, размер файла
- `public_key` – *str*, публичный ключ
- `sha256` – *str*, SHA256 хэш
- `md5` – *str*, MD5 хэш
- `embedded` – *ResourceListObject*, список вложенных ресурсов
- `name` – *str*, имя файла
- `exif` – *EXIFObject*, метаданные EXIF
- `resource_id` – *str*, идентификатор ресурса

- `custom_properties` – *dict*, пользовательские свойства ресурса
- `public_url` – *str*, публичный URL
- `share` – *ShareInfoObject*, информация об общей папке
- `modified` – `datetime.datetime`, дата последнего изменения
- `created` – `datetime.datetime`, дата создания
- `photoslice_time` – `datetime.datetime`, дата создания фото/видео
- `mime_type` – *str*, MIME-тип
- `path` – *str*, путь к ресурсу
- `preview` – *str*, URL превью файла
- `comment_ids` – *CommentIDsObject*, идентификаторы комментариев
- `type` – *str*, тип («file» или «dir»)
- `media_type` – *str*, тип файла, согласно Яндекс.Диску
- `revision` – *int*, ревизия Яндекс.Диска на момент последнего изменения

`class yadisk_async.objects.resources.ResourceUploadLinkObject(`*resource_upload_link=None*`)`

Базовые классы: *yadisk_async.objects.resources.LinkObject*

Ссылка для загрузки файла.

Параметры `resource_upload_link` – *dict* или *None*

Переменные

- `operation_id` – *str*, идентификатор операции по загрузке файла
- `href` – *str*, URL ссылки
- `method` – *str*, HTTP метод
- `templated` – *bool*, признак шаблонизированного URL

`class yadisk_async.objects.resources.ShareInfoObject(`*share_info=None*`)`

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект информации об общей папке.

Параметры `share_info` – *dict* или *None*

Переменные

- `is_root` – *bool*, признак того, что папка является корневой
- `is_owned` – *bool*, признак того, что пользователь является владельцем этой папки
- `rights` – *str*, права доступа

`class yadisk_async.objects.resources.PublicResourceObject(`*public_resource=None*`)`

Базовые классы: *yadisk_async.objects.resources.ResourceObject*

Объект публичного ресурса.

Параметры `resource` – *dict* или *None*

Переменные

- `antivirus_status` – *str*, статус проверки антивирусом

- `file` – *str*, URL для скачивания файла
- `size` – *int*, размер файла
- `public_key` – *str*, публичный ключ
- `sha256` – *str*, SHA256 хэш
- `md5` – *str*, MD5 хэш
- `embedded` – *PublicResourceObject*, список вложенных ресурсов
- `name` – *str*, имя файла
- `exif` – *EXIFObject*, метаданные EXIF
- `resource_id` – *str*, идентификатор ресурса
- `custom_properties` – *dict*, пользовательские свойства ресурса
- `public_url` – *str*, публичный URL
- `share` – *ShareInfoObject*, информация об общей папке
- `modified` – `datetime.datetime`, дата последнего изменения
- `created` – `datetime.datetime`, дата создания
- `photoslice_time` – `datetime.datetime`, дата создания фото/видео
- `mime_type` – *str*, MIME-тип
- `path` – *str*, путь к ресурсу
- `preview` – *str*, URL превью файла
- `comment_ids` – *CommentIDsObject*, идентификаторы комментариев
- `type` – *str*, тип («file» или «dir»)
- `media_type` – *str*, тип файла, согласно Яндекс.Диску
- `revision` – *int*, ревизия Яндекс.Диска на момент последнего изменения
- `view_count` – *int*, количество просмотров публичного ресурса
- `owner` – *UserPublicInfoObject*, владелец публичного ресурса

`class yadisk_async.objects.resources.PublicResourceListObject(`*public_resource_list=None*`)`

Базовые классы: *yadisk_async.objects.resources.ResourceListObject*

Список публичных ресурсов.

Параметры `public_resource_list` – *dict* или *None*

Переменные

- `sort` – *str*, тип сортировки
- `items` – *list*, список ресурсов (*ResourceObject*)
- `limit` – *int*, максимальное число элементов в списке
- `offset` – *int*, отступ от начала списка
- `path` – *str*, путь к папке, содержащей элементы списка
- `total` – *int*, количество элементов списка
- `public_key` – *str*, публичный ключ к ресурсу

class yadisk_async.objects.resources.TrashResourceObject(*trash_ resource=None*)

Базовые классы: *yadisk_async.objects.resources.ResourceObject*

Объект ресурса корзины.

Параметры `trash_resource` – *dict* или *None*

Переменные

- `antivirus_status` – *str*, статус проверки антивирусом
- `file` – *str*, URL для скачивания файла
- `size` – *int*, размер файла
- `public_key` – *str*, публичный ключ
- `sha256` – *str*, SHA256 хэш
- `md5` – *str*, MD5 хэш
- `embedded` – *ResourceListObject*, список вложенных ресурсов
- `name` – *str*, имя файла
- `exif` – *EXIFObject*, метаданные EXIF
- `resource_id` – *str*, идентификатор ресурса
- `custom_properties` – *dict*, пользовательские свойства ресурса
- `public_url` – *str*, публичный URL
- `share` – *ShareInfoObject*, информация об общей папке
- `modified` – `datetime.datetime`, дата последнего изменения
- `created` – `datetime.datetime`, дата создания
- `photoslice_time` – `datetime.datetime`, дата создания фото/видео
- `mime_type` – *str*, MIME-тип
- `path` – *str*, путь к ресурсу
- `preview` – *str*, URL превью файла
- `comment_ids` – *CommentIDsObject*, идентификаторы комментариев
- `type` – *str*, тип («file» или «dir»)
- `media_type` – *str*, тип файла, согласно Яндекс.Диску
- `revision` – *int*, ревизия Яндекс.Диска на момент последнего изменения
- `origin_path` – *str*, оригинальный путь
- `deleted` – `datetime.datetime`, дата удаления

class yadisk_async.objects.resources.TrashResourceListObject(*trash_ resource_ list=None*)

Базовые классы: *yadisk_async.objects.resources.ResourceListObject*

Список ресурсов корзины.

Параметры `trash_resource_list` – *dict* или *None*

Переменные

- `sort` – *str*, тип сортировки

- `items` – *list*, список ресурсов (*TrashResourceObject*)
- `limit` – *int*, максимальное число элементов в списке
- `offset` – *int*, отступ от начала списка
- `path` – *str*, путь к папке, содержащей элементы списка
- `total` – *int*, количество элементов списка

`class yadisk_async.objects.operations.OperationStatusObject(operation_status=None)`

Базовые классы: `yadisk_async.objects.yadisk_object.YaDiskObject`

Объект статуса операции.

Параметры `operation_status` – *dict* или *None*

Переменные

- `type` – *str*, тип операции
- `status` – *str*, статус операции
- `operation_id` – *str*, идентификатор операции
- `link` – *LinkObject*, ссылка на операцию
- `data` – *dict*, другая информация об операции

2.5 Низкоуровневый API

2.5.1 Вспомогательные средства

`yadisk_async.utils.get_exception(response)`

Возвращает объект исключения, основываясь на ответе (подразумевается, что запрос не удался).

Параметры `response` – объект `aiohttp.ClientResponse`

Результат Объект исключения, подкласс *YaDiskError*

`yadisk_async.utils.auto_retry(func, n_retries=None, retry_interval=None)`

Выполняет запрос с автоматическими повторными попытками. Повторная попытка может быть вызвана `aiohttp.ClientError` или *RetriableYaDiskError*.

Параметры

- `func` – Функция, подлежащая исполнению, не должна требовать аргументов
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – *int* или *float*, задержка между повторными попытками в секундах

Результат Значение, возвращаемое func()

2.5.2 Функции

`yadisk_async.functions.auth.check_token(session, **kwargs)`

Проверяет, действителен ли токен.

Параметры `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками

Результат *bool*

yadisk_async.functions.auth.get_auth_url(*client_id*, ***kwargs*)

Получает URL для аутентификации для пользователя.

Параметры

- `client_id` – идентификатор приложения
- `type` – тип ответа («code», чтобы получить код подтверждения или «token», чтобы получить токен автоматически)
- `device_id` – уникальный идентификатор устройства, от 6 до 50 символов
- `device_name` – имя устройства, не более 100 символов
- `display` – указывает использовать облегчённую вёрстку, обрабатывает только «popup», остальные значения игнорируются
- `login_hint` – username или email аккаунта, для которого будет получен токен
- `scope` – список разрешений для приложения
- `optional_scope` – список опциональных разрешений для приложения
- `force_confirm` – Если *True*, пользователь должен будет разрешить доступ к аккаунту, даже если он уже это сделал до этого
- `state` – Строка состояния, которую Яндекс.OAuth возвращает без изменений (<= 1024 символов)

Результат URL для аутентификации

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

Получает URL для получения пользователем кода подтверждения. Он может быть использован для получения токена.

Параметры

- `client_id` – идентификатор приложения
- `device_id` – уникальный идентификатор устройства, от 6 до 50 символов
- `device_name` – имя устройства, не более 100 символов
- `display` – указывает использовать облегчённую вёрстку, обрабатывает только «popup», остальные значения игнорируются
- `login_hint` – username или email аккаунта, для которого будет получен токен
- `scope` – список разрешений для приложения
- `optional_scope` – список опциональных разрешений для приложения
- `force_confirm` – Если *True*, пользователь должен будет разрешить доступ к аккаунту, даже если он уже это сделал до этого
- `state` – Строка состояния, которую Яндекс.OAuth возвращает без изменений (<= 1024 символов)

Результат URL для аутентификации

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

Получает новый токен.

Параметры

- `code` – код подтверждения

- `client_id` – идентификатор приложения
- `client_secret` – пароль приложения
- `device_id` – уникальный идентификатор устройства (между 6 и 50 символами)
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TokenObject*

`yadisk_async.functions.auth.refresh_token(`*refresh_token*, *client_id*, *client_secret*, ***kwargs*`)`
Обновляет существующий токен.

Параметры

- `refresh_token` – refresh-токен, полученный вместе с токеном
- `client_id` – идентификатор приложения
- `client_secret` – пароль приложения
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TokenObject*

`yadisk_async.functions.auth.revoke_token(`*token*, *client_id*, *client_secret*, ***kwargs*`)`
Отзывает токен.

Параметры

- `token` – токен, подлежащий отзыву
- `client_id` – идентификатор приложения
- `client_secret` – пароль приложения
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TokenRevokeStatusObject*

`yadisk_async.functions.disk.get_disk_info(`*session*, ***kwargs*`)`
Получает информацию о диске.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса

- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *DiskInfoObject*

`yadisk_async.functions.resources.copy(`*session*, *src_path*, *dst_path*, ***kwargs*`)`

Копирует *src_path* в *dst_path*. Если операция выполняется асинхронно, возвращает ссылку на операцию, иначе, возвращает ссылку на новый ресурс.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `src_path` – исходный путь
- `dst_path` – путь назначения
- `overwrite` – если *True*, путь назначения может быть перезаписан, иначе будет вызвана ошибка
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

`yadisk_async.functions.resources.download(`*session*, *src_path*, *file_or_path*, ***kwargs*`)`

Скачивает файл.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `src_path` – исходный путь
- `path_or_file` – путь назначения или файл-подобный объект
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`yadisk_async.functions.resources.exists(`*session*, *path*, ***kwargs*`)`

Проверяет, существует ли *path*.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса

- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`yadisk_async.functions.resources.get_download_link(`*session*, *path*, ***kwargs*`)`
Получает ссылку на скачивание файла (или папки).

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

`yadisk_async.functions.resources.get_meta(`*session*, *path*, ***kwargs*`)`
Получает мета-информацию о ресурсе.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *ResourceObject*

`yadisk_async.functions.resources.get_type(`*session*, *path*, ***kwargs*`)`
Получает тип ресурса

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

`yadisk_async.functions.resources.get_upload_link(`*session*, *path*, ***kwargs*`)`
Получает ссылку для загрузки файла на диск при помощи PUT запроса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь назначения
- `overwrite` – *bool*, определяет, перезаписывать путь назначения или нет
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

`yadisk_async.functions.resources.is_dir(`*session*, *path*, ***kwargs*`)`
Проверяет, является ли *path* папкой.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

`yadisk_async.functions.resources.is_file(`*session*, *path*, ***kwargs*`)`
Проверяет, является ли *path* файлом.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу

- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является файлом, *False*, в остальных случаях (даже если ресурс не существует)

`yadisk_async.functions.resources.listdir(`*session*, *path*, ***kwargs*`)`
Получает содержимое *path*.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к папке
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *ResourceObject*

`yadisk_async.functions.resources.mkdir(`*session*, *path*, ***kwargs*`)`
Создаёт новую папку.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к папке, подлежащей созданию
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*

`yadisk_async.functions.resources.remove(`*session*, *path*, ***kwargs*`)`
Удаляет ресурс.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к удаляемому ресурсу
- `permanently` – если *True*, ресурс будет безвозвратно удалён, иначе ресурс будет перемещён в корзину
- `md5` – *str*, MD5 хэш файла, подлежащего удалению
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*, если операция выполняется асинхронно, иначе *None*

`yadisk_async.functions.resources.upload(`*session*, *file_or_path*, *dst_path*, ***kwargs*`)`

Загружает файл на диск.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `file_or_path` – путь к файлу или файл-подобный объект для загрузки
- `dst_path` – путь назначения
- `overwrite` – если *True*, путь назначения может быть перезаписан, иначе будет вызвана ошибка
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`yadisk_async.functions.resources.get_trash_meta(`*session*, *path*, ***kwargs*`)`

Получает мета-информацию о ресурсе корзины.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `fields` – список ключей, которые будут включены в ответ

- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *TrashResourceObject*

yadisk_async.functions.resources.trash_exists(*session*, *path*, ***kwargs*)
Проверяет, существует ли *path* в корзине.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

yadisk_async.functions.resources.restore_trash(*session*, *path*, *dst_path=None*, ***kwargs*)
Восстанавливает ресурс корзины. Возвращает ссылку на новый ресурс или ссылку на асинхронную операцию.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к восстанавливаему ресурсу
- `dst_path` – путь назначения
- `overwrite` – *bool*, определяет может ли путь назначения быть перезаписан
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

yadisk_async.functions.resources.move(*session*, *src_path*, *dst_path*, ***kwargs*)
Перемещает *src_path* в *dst_path*.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `src_path` – исходный путь, подлежащий перемещению

- `dst_path` – путь назначения
- `overwrite` – *bool*, определяет, перезаписывать путь назначения или нет
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

`yadisk_async.functions.resources.remove_trash(`*session*, *path*, ***kwargs*`)`
Удаляет ресурс корзины.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины, подлежащий удалению
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *OperationLinkObject*, если операция выполняется асинхронно, иначе *None*

`yadisk_async.functions.resources.publish(`*session*, *path*, ***kwargs*`)`
Делает ресурс публичным.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к публикуемому ресурсу
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*, ссылка на ресурс

`yadisk_async.functions.resources.unpublish(`*session*, *path*, ***kwargs*`)`
Делает публичный ресурс приватным.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу, подлежащему депубликации
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject*

yadisk_async.functions.resources.save_to_disk(*session*, *public_key*, ***kwargs*)

Сохраняет публичный ресурс на диск. Возвращает ссылку на операцию, если сохранение выполняется асинхронно, или возвращает ссылку на ресурс.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `name` – имя файла/папки, под которым будет сохранён ресурс
- `path` – путь к копируемому ресурсу в публичной папке
- `save_path` – путь к папке назначения (загрузки по умолчанию)
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *LinkObject* или *OperationLinkObject*

yadisk_async.functions.resources.get_public_meta(*session*, *public_key*, ***kwargs*)

Получает мета-информацию о публичном ресурсе.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки. Указывая ключ опубликованной папки через *public_key*, вы можете запросить метаинформацию любого ресурса внутри неё.
- `offset` – отступ от начала списка вложенных ресурсов
- `limit` – максимальное количество элементов списка вложенных ресурсов
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла

- `preview_crop` – *bool*, разрешить обрезку превью
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *PublicResourceObject*

`yadisk_async.functions.resources.public_exists(`*session*, *public_key*, ***kwargs*`)`
Проверяет, существует ли публичный ресурс.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *bool*

`yadisk_async.functions.resources.public_listdir(`*session*, *public_key*, ***kwargs*`)`
Получает содержимое публичной папки.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу в публичной папке. Указывая ключ опубликованной папки через *public_key*, вы можете запросить содержимое любой вложенной папки.
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *PublicResourceObject*

yadisk_async.functions.resources.get_public_type(*session*, *public_key*, ***kwargs*)
Получает тип публичного ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

yadisk_async.functions.resources.is_public_dir(*session*, *public_key*, ***kwargs*)
Проверяет, является ли публичный ресурс публичной папкой.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *public_key* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

yadisk_async.functions.resources.is_public_file(*session*, *public_key*, ***kwargs*)
Проверяет, является ли публичный ресурс публичным файлом.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *public_key* является файлом, *False*, в остальных случаях (даже если ресурс не существует)

yadisk_async.functions.resources.trash_listdir(*session*, *path*, ***kwargs*)
Получает содержимое папки в корзине.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к папке в корзине
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *TrashResourceObject*

yadisk_async.functions.resources.get_trash_type(*session*, *path*, ***kwargs*)
Получает тип ресурса корзины.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат «file» или «dir»

yadisk_async.functions.resources.is_trash_dir(*session*, *path*, ***kwargs*)
Проверяет, является ли *path* папкой в корзине.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является папкой, *False*, в остальных случаях (даже если ресурс не существует)

yadisk_async.functions.resources.is_trash_file(*session*, *path*, ***kwargs*)
Проверяет, является ли *path* файлом в корзине.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу корзины
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *True*, если *path* является файлом, *False*, в остальных случаях (даже если ресурс не существует)

yadisk_async.functions.resources.get_public_resources(*session*, ***kwargs*)
Получает список публичных ресурсов.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `offset` – отступ от начала списка
- `limit` – максимальное число элементов в списке
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `type` – фильтр по типу ресурса («file» или «dir»)
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *PublicResourcesListObject*

yadisk_async.functions.resources.patch(*session*, *path*, *properties*, ***kwargs*)
Обновляет пользовательские свойства ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `properties` – *dict*, новые пользовательские свойства ресурса
- `fields` – список ключей, которые будут включены в ответ

- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *ResourceObject*

`yadisk_async.functions.resources.get_files(`*session*, ***kwargs*`)`

Получить плоский список всех файлов (без папок).

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `offset` – отступ от начала списка
- `limit` – максимальное количество элементов списка
- `media_type` – тип файлов, которые будут включены в список
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *ResourceObject*

`yadisk_async.functions.resources.get_last_uploaded(`*session*, ***kwargs*`)`

Получает список последних загруженных файлов, отсортированный по дате загрузки.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `limit` – максимальное число элементов в списке
- `media_type` – тип файлов, которые будут включены в список
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат генератор *LastUploadedResourceListObject*

yadisk_async.functions.resources.upload_url(*session*, *url*, *path*, ***kwargs*)
Загружает файл на диск по URL.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `url` – исходный URL
- `path` – путь назначения
- `disable_redirects` – *bool*, запретить делать перенаправления
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *OperationLinkObject*, ссылка на асинхронную операцию

yadisk_async.functions.resources.get_public_download_link(*session*, *public_key*, ***kwargs*)
Получает ссылку на скачивание публичного ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

yadisk_async.functions.resources.download_public(*session*, *public_key*, *file_or_path*, ***kwargs*)
Скачивает публичный ресурс.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к публичному ресурсу
- `file_or_path` – путь назначения или файл-подобный объект
- `path` – относительный путь к ресурсу внутри публичной папки
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса

- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

`yadisk_async.functions.operations.get_operation_status(`*session*, *operation_id*, ***kwargs*`)`
Получает статус операции.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `operation_id` – идентификатор операции или ссылка на нее
- `fields` – список ключей, которые будут включены в ответ
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах

Результат *str*

2.5.3 Объекты запросов к API

`class yadisk_async.api.APIRequest(`*session*, *args*, ***kwargs*`)`
Базовый класс для всех объектов запросов к REST API.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `args` – *dict*, аргументы, которые будут перданы в *process_args*
- `timeout` – *float* или *tuple*, таймаут запроса
- `headers` – *dict* или *None*, дополнительные заголовки запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `retry_interval` – задержка между повторными попытками в секундах
- `kwargs` – другие аргументы для `aiohttp.ClientSession.request`

Переменные

- `url` – *str*, URL запроса
- `method` – *str*, метод запроса
- `content_type` – *str*, заголовок Content-Type («application/x-www-form-urlencoded» по умолчанию)
- `timeout` – *float* или *tuple*, таймаут запроса
- `n_retries` – *int*, максимальное число повторных попыток запроса
- `success_codes` – *list*-подобный, список кодов ответов, означающих успех запроса
- `retry_interval` – *float*, задержка между повторными попытками в секундах

`process()`
Обрабатывает ответ.

Результат зависит от *self.process_json()*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`send()`

Отправляет запрос

Результат `aiohttp.ClientResponse` (*self.response*)

`class yadisk_async.api.auth.RefreshTokenRequest(session, refresh_token, client_id, client_secret, **kwargs)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для обновления существующего токена.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `refresh_token` – refresh-токен, полученный вместе с токеном
- `client_id` – идентификатор приложения
- `client_secret` – пароль приложения

Результат *TokenObject*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.auth.RevokeTokenRequest(session, token, client_id, client_secret, **kwargs)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для отзыва токена.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `token` – токен, подлежащий отзыву
- `client_id` – идентификатор приложения
- `client_secret` – пароль приложения

Результат *TokenRevokeStatusObject*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.auth.GetTokenRequest(*session*, *code*, *client_id*, *client_secret*, *device_id=None*, *device_name=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения токена.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- code – код подтверждения
- client_id – идентификатор приложения
- client_secret – пароль приложения
- device_id – уникальный идентификатор устройства (между 6 и 50 символами)

Результат *TokenObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.disk.DiskInfoRequest(*session*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения информации о диске.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- fields – список ключей, которые будут включены в ответ

Результат *DiskInfoObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.resources.GetPublicResourcesRequest(*session*, *offset=0*, *limit=20*, *preview_size=None*, *preview_crop=None*, *type=None*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения списка публичных ресурсов.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- offset – отступ от начала списка

- `limit` – максимальное число элементов в списке
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `type` – фильтр по типу ресурса («file» или «dir»)
- `fields` – список ключей, которые будут включены в ответ

Результат *PublicResourcesListObject*

`process_json(`*js*`)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.resources.UnpublishRequest(`*session*, *path*, *fields=None*, ***kwargs*`)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для того, чтобы сделать публичный ресурс приватным.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу, подлежащему депубликации
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject*

`process_json(`*js*`)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.resources.GetDownloadLinkRequest(`*session*, *path*, *fields=None*, ***kwargs*`)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для получения ссылки на скачивание ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к скачиваемому ресурсу
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject*

`process_json(`*js*`)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получния мета-информации о ресурсе корзины.

Параметры

- path – путь к ресурсу корзины
- limit – количество ресурсов в папке, которые будут включены в ответ
- offset – количество ресурсов в папке, которые будут пропущены
- preview_size – размер превью файла
- preview_crop – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- sort – *str*, поле используемое для сортировки вложенных ресурсов
- fields – список ключей, которые будут включены в ответ

Результат *TrashResourceObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.resources.RestoreTrashRequest(*session*, *path*, *dst_path=None*, *force_async=False*, *overwrite=False*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для восстановления мусора.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- path – путь к восстанавливаему ресурсу
- dst_path – путь назначения
- force_async – заставляет выполнять операцию асинхронно
- overwrite – *bool*, определяет может ли путь назначения быть перезаписан
- fields – список ключей, которые будут включены в ответ

Результат *LinkObject* или *OperationLinkObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для удаления ресурса корзины.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- path – путь к ресурсу корзины, подлежащий удалению
- force_async – заставляет выполнять операцию асинхронно
- fields – список ключей, которые будут включены в ответ

Результат *OperationLinkObject* или *None*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения списка последних загруженных файлов, отсортированного по дате загрузки.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- limit – максимальное число элементов в списке
- media_type – тип файлов, которые будут включены в список
- preview_size – размер превью файла
- preview_crop – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- fields – список ключей, которые будут включены в ответ

Результат *LastUploadedResourceListObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.resources.CopyRequest(*session*, *src_path*, *dst_path*, *overwrite=False*, *force_async=False*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос копирования файла или папки.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `src_path` – исходный путь
- `dst_path` – путь назначения
- `overwrite` – если *True*, путь назначения может быть перезаписан, иначе будет вызвана ошибка
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject* или *OperationLinkObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.resources.GetMetaRequest(`*session*, *path*, *limit=None*, *offset=None*, *preview_size=None*, *preview_crop=None*, *sort=None*, *fields=None*, ***kwargs*`)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для получния мета-информации о ресурсе.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `limit` – количество ресурсов в папке, которые будут включены в ответ
- `offset` – количество ресурсов в папке, которые будут пропущены
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `fields` – список ключей, которые будут включены в ответ

Результат *ResourceObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для получения ссылки для загрузки ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь назначения для загрузки файла
- `overwrite` – *bool*, определяет, перезаписывать путь назначения или нет
- `fields` – список ключей, которые будут включены в ответ

Результат *ResourceUploadLinkObject*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для создания новой папки.

Параметры

- `path` – путь к папке, подлежащей созданию
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для того, чтобы сделать ресурс публичным.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к публикуемому ресурсу
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject*

`process_json(js)`
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для загрузки файла по URL.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- url – исходный URL
- path – путь назначения
- disable_redirects – *bool*, запретить делать перенаправления
- fields – список ключей, которые будут включены в ответ

Результат *OperationLinkObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для удаления ресурса.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- path – путь к удаляемому ресурсу
- permanently – если *True*, ресурс будет безвозвратно удалён, иначе ресурс будет перемещён в корзину
- force_async – заставляет выполнять операцию асинхронно
- md5 – *str*, MD5 хэш файла, подлежащего удалению
- fields – список ключей, которые будут включены в ответ

Результат *OperationLinkObject* или *None*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для сохранения публичного ресурса на диск.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к ресурсу
- `name` – имя файла/папки, под которым будет сохранён ресурс
- `path` – путь к копируемому ресурсу в публичной папке
- `save_path` – путь к папке назначения (загрузки по умолчанию)
- `force_async` – заставляет выполнять операцию асинхронно
- `fields` – список ключей, которые будут включены в ответ

Результат *LinkObject* или *OperationLinkObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для получния мета-информации о публичном ресурсе.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `public_key` – публичный ключ или URL к ресурсу
- `path` – относительный путь к ресурсу внутри публичной папки. Указывая ключ опубликованной папки через *public_key*, вы можете запросить метаинформацию любого ресурса внутри неё.
- `offset` – отступ от начала списка вложенных ресурсов
- `limit` – максимальное количество элементов списка вложенных ресурсов
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, разрешить обрезку превью
- `fields` – список ключей, которые будут включены в ответ

Результат *PublicResourceObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

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

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения ссылки на скачивание публичного ресурса.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- public_key – публичный ключ или URL к ресурсу
- path – относительный путь к ресурсу внутри публичной папки
- fields – список ключей, которые будут включены в ответ

Результат *LinkObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.resources.MoveRequest(*session*, *src_path*, *dst_path*, *force_async=False*, *overwrite=False*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для перемещения ресурса.

Параметры

- session – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- src_path – исходный путь, подлежащий перемещению
- dst_path – путь назначения
- force_async – заставляет выполнять операцию асинхронно
- overwrite – *bool*, определяет, перезаписывать путь назначения или нет
- fields – список ключей, которые будут включены в ответ

Результат *OperationLinkObject* или *LinkObject*

process_json(*js*)

Обрабатывает JSON ответ.

Параметры js – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

class yadisk_async.api.resources.FilesRequest(*session*, *offset=0*, *limit=20*, *media_type=None*, *preview_size=None*, *preview_crop=None*, *sort=None*, *fields=None*, ***kwargs*)

Базовые классы: yadisk_async.api.api_request.APIRequest

Запрос для получения плоского списка всех файлов.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `offset` – отступ от начала списка
- `limit` – максимальное количество элементов списка
- `media_type` – тип файлов, которые будут включены в список
- `sort` – *str*, поле используемое для сортировки вложенных ресурсов
- `preview_size` – размер превью файла
- `preview_crop` – *bool*, обрезает превью согласно размеру, заданному в *preview_size*
- `fields` – список ключей, которые будут включены в ответ

Результат *FilesResourceListObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.resources.PatchRequest(`*session*, *path*, *properties*, *fields=None*, ***kwargs*`)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для обновления пользовательских свойств ресурса.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `path` – путь к ресурсу
- `properties` – *dict*, новые пользовательские свойства ресурса
- `fields` – список ключей, которые будут включены в ответ

Результат *ResourceObject*

`process_json(`*js*`)`

Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

`class yadisk_async.api.operations.GetOperationStatusRequest(`*session*, *operation_id*, *fields=None*, ***kwargs*`)`

Базовые классы: `yadisk_async.api.api_request.APIRequest`

Запрос для получения статуса операции.

Параметры

- `session` – объект *yadisk_async.session.SessionWithHeaders* с приготовленными заголовками
- `operation_id` – идентификатор операции или ссылка на нее
- `fields` – список ключей, которые будут включены в ответ

Результат *OperationStatusObject*

process_json(*js*)
Обрабатывает JSON ответ.

Параметры `js` – *dict*, JSON ответ

Результат обработанный ответ, может быть что угодно

Глава 3

История изменений

- **Release 1.3.0 (2019-07-06)**
  - Реализована поддержка *async/await*
  - Библиотека была переименована из yadisk в *yadisk-async*

Следующие релизы относятся к оригинальной библиотеке yadisk:

- **Release 1.2.14 (2019-03-26)**
  - Исправлена ошибка *TypeError* в функциях *get_public_** при использовании с параметром *path* (issue #7)
  - Добавлен аттрибут *unlimited_autoupload_enabled* для *DiskInfoObject*
- **Release 1.2.13 (2019-02-23)**
  - Добавлен параметр *md5* для *remove()*
  - Добавлен *UserPublicInfoObject*
  - Добавлен аттрибут *country* для *UserObject*
  - Добавлен аттрибут *photoslice_time* для *ResourceObject*, *PublicResourceObject* и *TrashResourceObject*
- **Release 1.2.12 (2018-10-11)**
  - Исправлен баг: не работает параметр *fields* в *listdir()* (issue #4)
- **Release 1.2.11 (2018-06-30)**
  - Добавлен недостающий параметр *sort* для *get_meta()*
  - Добавлены аттрибуты *file* и *antivirus_status* для *ResourceObject*, *PublicResourceObject* и *TrashResourceObject*
  - Добавлен параметр *headers*
  - Исправлена опечатка в *download()* и *download_public()* (issue #2)
  - Убран параметр **args*

- **Release 1.2.10 (2018-06-14)**
  - Исправлено поведение *timeout=None*. *None* должен означать „без таймаута“, но в предыдущих версиях значение *None* было синонимично со стандартным таймаутом.
- **Release 1.2.9 (2018-04-28)**
  - Изменена лицензия на LGPLv3 (см. *COPYING* и *COPYING.lesser*)
  - Другие изменения информации о пакете
- **Release 1.2.8 (2018-04-17)**
  - Исправлено несколько опечаток: у *PublicResourceListObject.items* и *TrashResourceListObject.items* были неправильные типы данных
  - Псевдонимы полей в параметре *fields* заменяются при выполнении запросов API (например, *embedded -> _embedded*)
- **Release 1.2.7 (2018-04-15)**
  - Исправлен баг перемотки файла при загрузке/скачивании после повторной попытки
- **Release 1.2.6 (2018-04-13)**
  - Теперь объекты сессий *requests* кэшируются, чтобы их можно было переиспользовать (иногда может существенно ускорить выполнение запросов)
  - *keep-alive* отключается при загрузке/скачивании файлов по умолчанию
- **Release 1.2.5 (2018-03-31)**
  - Исправлен баг (ошибка на единицу) в *utils.auto_retry()* (иногда мог вызвать *AttributeError*)
  - Повторные попытки применяются для *upload()*, *download()* и *download_public()* целиком
  - Задано *stream=True* для *download()* и *download_public()*
  - Другие мелкие исправления
- **Release 1.2.4 (2018-02-19)**
  - Исправлена опечатка (*TokenObject.exprires_in -> TokenObject.expires_in*)
- **Release 1.2.3 (2018-01-20)**
  - Исправлено *TypeError* при вызове *WrongResourceTypeError*
- **Release 1.2.2 (2018-01-19)**
  - *refresh_token()* больше не требует валидный или пустой токен.
- **Release 1.2.1 (2018-01-14)**
  - Исправлена неработоспособность повторных попыток.
- **Release 1.2.0 (2018-01-14)**
  - Исправлено использование *n_retries=0* в *upload()*, *download()* и *download_public()*
  - *upload()*, *download()* и *download_public()* больше не возвращают ничего (см. документацию)
  - Добавлен модуль *utils* (см. документацию)
  - Добавлены *RetriableYaDiskError*, *WrongResourceTypeError*, *BadGatewayError* и *GatewayTimeoutError*
  - *listdir()* теперь вызывает *WrongResourceTypeError* вместо *NotADirectoryError*
- **Release 1.1.1 (2017-12-29)**

  - Исправлена обработка аргументов в *upload()*, *download()* и *download_public()*. До этого использование *n_retries* и *retry_interval* вызывало исключение (*TypeError*).

- **Release 1.1.0 (2017-12-27)**
  - Усовершенствованные исключения (см. документацию)
  - Добавлена поддержка параметра *force_async*
  - Мелкие исправления багов
- **Release 1.0.8 (2017-11-29)**
  - Исправлен ещё один баг в *listdir()*
- **Release 1.0.7 (2017-11-04)**
  - Добавлен *install_requires* в *setup.py*
- **Release 1.0.6 (2017-11-04)**
  - Некоторые функции теперь возвращают *OperationLinkObject*
- **Release 1.0.5 (2017-10-29)**
  - Исправлен *setup.py*, теперь исключает тесты
- **Release 1.0.4 (2017-10-23)**
  - Исправлены баги в *upload*, *download* и *listdir*
  - Значение по-умолчанию *limit* в *listdir* установлено в *10000*
- **Release 1.0.3 (2017-10-22)**
  - Добавлен модуль *settings*
- **Release 1.0.2 (2017-10-19)**
  - Исправлена функция *get_code_url* (добавлены недостающие параметры)
- **Release 1.0.1 (2017-10-18)**
  - Исправлен серьёзный баг в *GetTokenRequest* (добавлен недостающий параметр)
- **Release 1.0.0 (2017-10-18)**
  - Первый релиз

Глава 4

Indices and tables

- genindex
- modindex
- search

Содержание модулей Python

y

I

L

M

R

S

T

U

W

Y