Общее описание работы с API

API представляет из себя набор REST запросов. Параметры могут кодироваться в URL, и теле запроса. В параметрах запроса передается только id сессии.

Все действия, кроме входа в систему доступны только авторизованных пользователям. Авторизация производится по JSON строке в теле запроса {"login":"...","password":"..."} (может содержать поле otp, если у клиента включена авторизация через google authenticator). После успешной авторизации выдается сессионный ключ на 24 часа, который должен присутствовать в любом последующем запросе в виде атрибута sessionid заголовка каждого запроса. При потоке запросов более 10 в 1 секунду все входящие запросы могут быть заблокированы на произвольное время, что зависит от степени нагрузки сервиса.

Структура REST запросов

Тип	URL	Описание
GET	/api/login	Пустой запрос, возвращает информацию о пользователе {login:"",fullName:"",description:"",security:[], sessionId:"",friends:{“login”:”name”,...}}
DELETE		Пустой запрос, принудительно закрывает сессию
POST		Запрос с данными {"login":"...","password":"..."}. Производит авторизацию в сервисе и выдачу сессии. captcha обязательна только в случае если был детектирован перебор пароля по данной учетной записи
POST	/api/resource/search	В теле запроса передаются фильтры. Возвращает все подходящие документы (максимум 1000). В теле запроса можно установить фильтры в формате { “dateFrom”:UTS “dateTo”:UTS “user”:”документы по пользователю” “limit”:макс количество документов “page”:следующие {limit} документы }, где UTS - UnixTimestamp с миллисекундами
GET	/api/resource/{id}	Получить информацию о документе по id

	/cdn/{src}/{hash}	Получить оригинал документа в формате PDF, src и hash
PUT	/api/resource/name/{id}	Переименовать документ по id, новое имя передается в теле. Имя обрезается до 128 символов.
DELETE	/api/resource/{id}	Переместить в корзину или удалить документ если он находится в корзине
PUT	/api/resource/restore/{id}	Восстановить документ по id из корзины
PUT	/api/resource/shareall/{id}	Включить доступ всем по ссылке для указанного по id документа. Ссылка на файл будет иметь вид: /share/{hash}{id}
DELETE		Выключить доступ всем по ссылке для указанного по id документа
PUT	/api/resource/tag/{id}	Назначить тэги (ярлыки) документу. В теле запроса передается число, каждый бит которого кодирует включение того или иного ярлыка
POST	/upload	Загрузить новый ресурс. Имя будет взято из параметра запроса filename, в ответ возвращает информацию о новом ресурсе.
GET	/api/share/{id}	Получить список тех, кому расшарен документ по id документа
POST		Расшарить документ, в теле запроса передается коммент + список emailов в формате {comment:"", emails:[]}
GET	/api/sign/{id}	Получить список подписей под по id
POST		Добавить подпись в документ, подпись в виде набора байт в Base64 передается в теле запроса
GET	/api/resource/withsign/{id}	Получить подписанный документ

Содержание структур, возвращаемые API

```
resource = {
      // Уникальный id документа
      "id":21,

      // Хеш от содержимого документа, может повторятся,
```

```
// если залить одинаковые данные
"hash":"875DPYO6AbafqGHJQ9Lu6H_cVyeRmcm6L_JLivAxAjo",

// Размер документа в байтах
"size":851691,

// Время создания документа
"time":1456966953394,

// Тип ресурса (0 - PDF, 1 - зашифрованный на пароле PDF)
"type":0,


// Префикс для доступа к ресурсу в данный момент это дата,
// но в будущем может быть заменен на произвольный набор
// для доступа к ресурсу URL формируется по правилу
// “https://paperless.com.ua/cdn/” + SRC + “/” + HASH
"src":"2018/03/03",

// Имя документа
"name":"Мой любимый документ для подписи",

// Логин автора
"author":"aleksandrgarashenko@gmail.com",

// Статус жизненного цикла документа
// CREATED = 0
// SHARED = 5
// SHARED_ALL = 6
// IN_TRASH = 10 (может суммироваться с теми, что меньше 10)
// DELETED = 20 (может суммироваться с теми, что меньше 10)
// Например, статус 16, значит что документ
// в корзине (IN_TRASH) и расшарен всем (SHARED_ALL)
"status":5,

// Набор меток, которые установил автор документа
// (доступен только автору документа)
// Каждый установленный бит в числе обозначает
```

```
    // установленный ярлык, например 5 -> 101, т.е. у документа
    // установлен 1 и 3 флаг из личного справочника ярлыков
    "tags":14,

    // Список людей, которым был расшарен документ со статусом
    // SEND_TO_UNKNOWN_USER = 0 пользователь не зарегистрирован
    // SEND_TO_KNOWN_USER = 1   еще не смотрел документ
    // VIEW_BY_USERS = 2        просмотрел документ
    // SIGNED_BY_USERS = 3      подписал документ
    // IN_TRASH = 10            удалил в корзину (статус может
    //                          добавляться к предыдущим, но
    //                         сейчас не используется)
    "shares":{"test@test.com":2}
}
```

Коды ошибок запросов :

4xx : ошибки с клиентскими данными

400 : неправильные данные

401 : неверный пароль или нет прав на действие у текущего пользователя

403 : доступ к данным запрещен, необходимо пройти авторизацию

404 :ресурс не найден

405 : данный метод недоступен в данном API

409: ресурс уже существует

415 : действие с ресурсом не поддерживается

423 : сессия истекла или не найдена, необходимо авторизоваться. При логине код значит необходимость добавления в запрос параметра otp (если у клиента включена аутентификация по одноразовому паролю).

429: слишком много повторяющихся запросов

5xx : ошибки сервера

500 : на сервере произошла неизвестная ошибка