System API

Системный API

Оглавление

Общие сведения

В документе описаны основные принципы взаимодействия клиента и сервера. Раскрыто понятие "Операция", рассмотрены свойства объектности операций а также затронуты некоторые стороны локализации системы.

Операции

Доступ к функционалу NGCP производится через набор "операций". Выполнение любой операции производится при помощи вызова `POST /api/operation/{operationName}` .
Аргументом вызова является тело следующего вида:

```
{
    "objectIds": [
        ...
    ],
    "args": {
        ...
    },
    "ignoreWarnings": {
        ...
    }
}
```

Здесь:

- `objectIds` это набор идентификаторов объектов в контексте которых выполняется запрошенная операция;
- `args` специфический набор аргументов для выполнения операции;
- `ignoreWarnings` набор предупреждений, которые требуется игнорировать при выполнении операции.

Возможны следующие ответы:

HTTP код	Описание	Результат
200	Запрос выполнен успешно	См. ниже.
404	Указанная операция не найдена	-
415	Если клиент ожидает получить JSON, но сервер хочет вернуть содержимое документа	ProblemDetail
500	Внутренняя ошибка сервиса	ProblemDetail

В случае успешного выполнения запроса (код ответа 200), результат выглядит следующим образом:

```
[
    {
        "objectIds": [],
        "preconditionMessages": {
            ...
        },
        "validationMessages": {
            ...
        },
        "actionMessages": {
            ...
        },
        "value": ...
    },
    ...
]
```

Здесь:

- `[i].objectIds` это набор идентификаторов объектов, к которым применим результат выполнения операции;
- `[i].preconditionMessages` сообщения предусловий;
- `[i].validationMessages` сообщения валидаций;
- `[i].actionMessages` сообщения действий;
- `[i].value` результат выполнения операции.

Форма результата выполнения зависит от операции, запуск которой был произведен.

"Объектность" операций

Все операции делятся на 3 условных типа.

- Необъектные -- это операции которые работают вне контекста какого-либо объекта. Поле `objectIds` в запросах на выполнение таких операций должно быть пустым или отсутствовать. Примером таких операций могут служить "Получение идентификаторов проектов" или "Получение задач пользователя".
- Объектные -- это операции которые работают в контексте одного объекта. Поле `objectIds` в запросах на выполнение таких операций должно содержать один и более идентификаторов. Если в поле `objectIds` передано более одного идентификатора, операция будет запущена индивидуально для каждого из переданных идентификаторов, с выполнением всех предусловий и проверок.
- Мульиобъектные -- это операции которые работают в контексте нескольких объектов. Поле `objectIds` в запросах на выполнение таких операций должно содержать один и более идентификаторов. Операция будет запущена один раз для всех объектов, независимо от количества переданных идентификаторов в поле `objectIds` .

В результате выполнения операции может быть один и более объектов. Количество объектов в результате зависит от того какой у операции тип, и от количества идентификаторов в поле `objectIds` .

- необъектные и мультиобъектные операции всегда возвращают один результат;
- объектные операции возвращают столько результатов, сколько идентификаторов передано в `objectIds` .

Сообщения предусловий

Если выполнение операции запрещено по результатам выполнения какого-либо предусловия, соответствующие сообщения появляются в поле `preconditionMessages` . Каждый ключ объекта отображает имя предусловия. Значение это список объектов вида:

```
{
    "status": "DISABLED",
    "message": ...
}
```

Если в ответе присутствует непустое поле `[i].preconditionMessages` , это значит что операция не была выполнена для объектов `[i].objectIds` . Причины указаны в прикрепленном сообщении. Пример ответа с провалами предусловий:

```
[
    {
        "objectIds": [
            "2fe62e17-7d6f-4388-81f1-8a7597e57e39",

```

```
            "0e4cb741-6159-4149-9d7d-88ba82b49a23"
        ],
        "preconditionMessages": { "isTaskOwner": [{ "status": "DENIED" }] },
        "validationMessages": {},
        "actionMessages": {},
        "value": null
    }
]
```

Сообщения валидации

Если пользователь некорректно заполнил атрибуты операции, то в результате выполнения операции в поле `validationMessages` появляются соответствующие сообщения. Каждый ключ объекта отображает имя действия, производящего валидацию. Значение это список объектов вида:

```
{
    "status": "ERROR",
    "message": ...
}
```

Если в ответе присутствует непустое поле `[i].validationMessages` , это значит что операция не была выполнена для объектов `[i].objectIds` . Причины указаны в прикрепленном сообщении.

Поле `status` может иметь значения `ERROR` или `WARNING` . Если поле `status` имеет значение `WARNING` , это значит что операция не завершилась с предупреждением, но можно перезапустить операцию заглушив это предупреждение. Для этого в тело запроса на выполнение операции требуется передать поле `ignoreWarnings` . Далее представлен пример запроса на выполнение операции с игнорированием предупреждений "MSG_REVIEW_CONSOLIDATION_SUSPENDED" от действия "reviewFinishConsolidation":

```
{
    "objectIds": [ ... ],
    "args": { ... },
    "ignoreWarnings": {
        "reviewFinishConsolidation": [ "MSG_REVIEW_CONSOLIDATION_SUSPENDED" ]
    }
}
```

Пример ответа с провалами валидации:

```
[
    {
        "objectIds": [
            "2fe62e17-7d6f-4388-81f1-8a7597e57e39",
            "0e4cb741-6159-4149-9d7d-88ba82b49a23"
        ],
        "preconditionMessages": {},
        "validationMessages": {
            "fetchUserTasks": [
                {
                    "status": "ERROR",
                    "message": {
                        "key": "MSG_INVALID_TASK_STATUS_ERROR",
                        "args": {
                            "type": "string",
                            "value": "ABRACADABRA"
                        }
                    }
                }
            ]
        },
        "actionMessages": {},
        "value": null
    }
]
```

Сообщения действий

Если в результате выполнения операции произошла ошибка, например ошибка состояния или другая непредвиденная ситуация, в поле `actionMessages` появляются соответствующие сообщения. Каждый ключ объекта отображает имя действия, в ходе выполнения которого произошла ошибка. Значение это список сообщений.

Если в ответе присутствует непустое поле `[i].actionMessages` , это значит что операция не была выполнена для объектов `[i].objectIds` . Причины указаны в прикрепленном сообщении. Пример ответа:

```
[
    {
        "objectIds": [
            "2fe62e17-7d6f-4388-81f1-8a7597e57e39",
            "0e4cb741-6159-4149-9d7d-88ba82b49a23"
        ],
        "preconditionMessages": {},
        "validationMessages": {},
        "actionMessages": {
            "fetchUserTasks": [
                {
                    "status": "ERROR",
                    "message": { "key": "MSG_OUT_OF_MEMORY_ERROR" }
                }
            ]
        },
        "value": null
    }
]
```

Получение списка доступных операций

Для определения доступности тех или иных операций в контексте выбранных объектов, требуется произвести специальный вызов, который для всех доступных в системе операций произведет фильтрацию по:

- типам объектов;
- ролям пользователей;
- предусловиям.

Запрос: `POST /api/operation/available`

В тело запроса требуется передать JSON массив с идентификаторами объектов, по которым проверяется доступность операций.

Возможны следующие ответы:

HTTP код	Описание	Результат
200	Запрос выполнен успешно	См. ниже.
500	Внутренняя ошибка сервиса	ProblemDetail

В случае успешного выполнения запроса (код ответа 200), результат выглядит следующим образом:


```
[
    {
        "objectIds": [
            "2fe62e17-7d6f-4388-81f1-8a7597e57e39"
        ],
        "operations": [
            { "name": "markTaskAs", "status": "AVAILABLE", "messages": []},
            { "name": "finishReviewComment", "status": "AVAILABLE", "messages": []}
        ]
    }, {
        "objectIds": [
            "0e4cb741-6159-4149-9d7d-88ba82b49a23"
        ],
        "operations": [
            { "name": "markTaskAs", "status": "AVAILABLE", "messages": []},
            { "name": "startReviewConsolidation", "status": "AVAILABLE", "messages": []}
        ]
    }, {
        "objectIds": [
            "2fe62e17-7d6f-4388-81f1-8a7597e57e39",
            "0e4cb741-6159-4149-9d7d-88ba82b49a23"
        ],
        "operations": [
            { "name": "markTaskAs", "status": "AVAILABLE", "messages": []}
        ]
    }
]
```

Здесь приведен пример ответа, при запросе списка доступных операций относительно двух задач. В результате имеется список из трех объектов:

- доступные объектные и мультиобъектные операции для первой задачи;
- доступные объектные и мультиобъектные операции для второй задачи;
- доступные мультиобъектные операции для первой и второй задач.

Поле `[i].operations[j].status` может иметь значения `AVAILABLE` или `DISABLED`. Если значение статуса `DISABLED`, это значит что контрол операции должен отображаться в интерфейсе, но быть недоступным. В поле `messages` записываются причины, по которым операция недоступна.

Получение реестра локализации атрибутов и сообщений

Система предоставляет информацию об именовании атрибутов для различных типов документов, а также системных сообщений. Для получения реестра локализации атрибутов требуется произвести соответствующий вызов.

Запрос: `GET /api/locregistry`

Ответ:

```
{
    "project": {
        "objectName": {
            "en": "Project Name",
            "ru": "Имя проекта"
        },
        "code": {
            "en": "Project Code",
            "ru": "Код проекта"
        },
        ...
    },
    "transmittal": {
        "objectName": {
            "en": "Transmittal Name",
            "ru": "Имя трансмиттала"
        },
        "objectNumber": {

```

```
            "en": "Transmittal Number",
            "ru": "Номер трансмиттала"
        },
        ...
    },
    "task": {
        "subject": {
            "en": "Subject",
            "ru": "Тема"
        },
        "description": {
            "en": "Description",
            "ru": "Описание"
        }
    },
    "groupName": {
        "main": {
            "en": "Main",
            "ru": "Основные"
        }
    },
    "message": {
        "MSG_TASK_REVIEW_COMMENT_SUBJECT": {
            "en": "Comment documents for process ''{0}''.",
            "ru": "Прокомментируйте документы для процесса ''{0}''"
        }
    }
    ...
}
```

Описание: запрос для получения реестра локализации атрибутов. Результат содержит исчерпывающие данные о наименованиях атрибутов системы. Каждый ключ результата описывает локализацию объектов соответствующего типа:

- `project` проекты;
- `documentType` типы документов;
- `document` документы;

- `comment` комментарии;
- `transmittalType` типы трансмитталов;
- `transmittal` трансмитталы;
- `processType` типы процессов;
- `process` процессы;
- `task` задачи;
- `bulkloadSheet` реестр массовой загрузки;
- `groupName` имена групп атрибутов;
- `message` сообщения.

Содержимое реестра локализации:
api_locregistry.json

Локализация сообщений

Иногда для пользователя требуется отобразить сообщения. Система в ответах на вызовы не передает текст сообщений. Вместо этого передается объект вида:

```
{
    "key": "MSG_TASK_REVIEW_CONSOLIDATE_SUBJECT",
    "args": [
        { "type": "string", "value": "ABC55990-DEF-GH-059" }
    ]
}
```

Здесь:

- `key` - это ключ сообщения;
- `args` - это набор аргументов, которые нужно подставить в сообщение.

Соответствия ключа сообщения и текста, соответствующего локали пользователя, представлены в реестре локализации атрибутов и сообщений. Формат сообщения соответствует формату из ECMA-402 . Для получения конечной строки для отображения пользователю предлагается использовать JS реализацию

Intl MessageFormat .

Каждый аргумент сообщения может иметь один из следующих типов:

- `string` - простая строка, подставляется как есть;
- `float` - число с плавающей запятой, может быть представлено в формате десятичной дроби, целого числа или экспоненциальном формате, например "13", "13.159", "1.3159e1";
- `integer` - целое число, например "1799";
- `time` - время, например "16:06:20.481564600";
- `date` - дата, например "2023-06-19";
- `datetime` - дата и время, например "2023-06-19T16:07:54.562619300";
- `datetime_os` - дата и время с часовым поясом, например "2023-06-19T16:08:40.891853+05:00";
- `datetime_tz` - дата и время с временной зоной, например "2023-06-19T16:10:36.997686+05:00[Asia/Yekaterinburg]";
- `instant` - момент времени, количество миллисекунд начиная с UNIX эпохи, например "1687173703371".

В приведенном выше примере, если ключу `MSG_TASK_REVIEW_CONSOLIDATE_SUBJECT` в реестре локализации соответствует строка

> Произведите консолидацию комментариев документов процесса "{0}"

то после преобразования конечное сообщение будет иметь вид

> Произведите консолидацию комментариев документов процесса 'ABC55990-DEF-GH-059'

Получение схемы системных типов

API шлюз предоставляет клиентам частичную схему данных, необходимую для отображения документов, процессов, трансмитталов и т.п. Если тот или иной атрибут объекта не представлен в схеме, то этот атрибут считается специальным и не предназначен для отображения, или должен отображаться/редактироваться специальным образом.

Запрос: `GET /api/typeschema` .

В результате запроса, клиент получает ответ в виде:


```
{
    "document": {
        "objectName": "string",
        "objectNumber": "string",
        "projectId": "projectId",
        "templateType": "string",
        "creationTemplateId": "documentTemplateId",
        "internalSysNumber": "string",
        "sequenceNumber": "integer",
        "objectTypeId": "documentTypeId",
        "chronicleId": "string",
        "version": "integer",
        "versionCounter": "string",
        "lastVersion": "boolean",
        "lastRevisionVersion": "boolean",
        "issueReason": "string",
        "revision": "string",
        "revisionDate": "instant",
        "status": "string",
        "reviewStatus": "string",
        "checkoutUserId": "userId",
        "creationDate": "instant",
        "author": "userId",
        "modifyDate": "instant",
        "modifyAuthor": "userId",
        "customer": "string",
        "originator": "string",
        "title": "string",
        "areaCode": "string",
        "packageNumber": "string",
        "packageCode": "string",
        "unitTitle": "string",
        "discipline": "string",
        "docSubtype": "string",
        "contractNumber": "string",
```

```
        "contractorDocNumber": "string",
        "reviewCode": "string",
        "stampName": "string",
        "stampDate": "string",
        "stampStatus": "string",
        "constructionStatus": "string"
    },
    "process": {
        "templateType": "enum (OBJECT, TEMPLATE_DRAFT, TEMPLATE_ACTIVE, TEMPLATE_INACTIVE)",
        "sequenceNumber": "integer",
        "objectTypeId": "string",
        "modifyDate": "instant",
        "documents": "documentId[]",
        "author": "userId",
        "objectNumber": "string",
        "description": "string",
        "project": "projectId",
        "creationDate": "instant",
        "objectType": "processTypeId",
        "modifyAuthor": "userId",
        "objectName": "string",
        "internalSysNumber": "string",
        "completionDate": "instant",
        "projectId": "string",
        "startDate": "instant",
        "status": "enum (NEW, STARTED, COMPLETED, ABORTED)"
    },
    ...
}
```

В данной схеме представлены следующие системные типы:

- `project` проекты;
- `documentType` типы документов;
- `document` документы;
- `comment` комментарии;

- `transmittalType` типы трансмитталов;
- `transmittal` трансмитталы;
- `processType` типы процессов;
- `process` процессы;
- `task` задачи;
- `groupName` имена групп атрибутов;
- `bulkloadSheet` реестр массовой загрузки.

Каждое значение, это соответствие атрибута и типа этого атрибута. Набор допустимых типов атрибутов:

- `string` - простая строка, подставляется как есть;
- `float` - число с плавающей запятой, может быть представлено в формате десятичной дроби, целого числа или экспоненциальном формате, например "13", "13.159", "1.3159e1";
- `integer` - целое число, например "1799";
- `time` - время, например "16:06:20.481564600";
- `date` - дата, например "2023-06-19";
- `datetime` - дата и время, например "2023-06-19T16:07:54.562619300";
- `instant` - момент времени, количество миллисекунд начиная с UNIX эпохи, например "1687173703371";
- `userId` - идентификатор пользователя системы;
- `projectId` - идентфификатор проекта;
- `documentId` - идентификатор документа;
- `documentTypeId` - идентификатор типа документов;
- `documentTemplateId` - идентификатор шаблона документа.

Массивы и списки задаются постфиксом `[]`, например `string[]` - массив строк.

Далее приведен список специальных атрибутов, которые не попадают в схему, клиент должен обрабатывать эти поля отдельно:

- для всех типов:

- additionalAttachments;
- autonumberConfig;
- attrsConfig
- content;
- objectType;
- project;

- для комментариев
  - parentComment;
- для трансмитталов
  - docTypes;
  - documents;
  - bo}ject;

@Deprecated api_typeschema.json
см. http://10.42.4.27:8908/api/typeschema