numi-api Documentation

Release 0.0.0

Praekelt Foundation

May 05, 2016

1 Overview **1**
1.1 Format 1
1.2 Verbs 1
1.3 Partial updates 1
1.4 Client errors 2
1.5 Read-only fields 3
1.6 Concepts 3
1.7 Permissions 5

2 Users **7**

3 Permissions **11**

4 Projects **13**

5 Dialogues **17**
5.1 Archiving dialogues 20

6 Revisions **21**
6.1 Ordering revisions 22

7 Releases **25**
7.1 Ordering releases 25

8 Channels **27**
8.1 Channel access 29

9 Providers **31**

10 Data Structures **33**
10.1 Users 33
10.2 Projects 35
10.3 Dialogues 36
10.4 Revisions 40
10.5 Releases 41
10.6 Permissions 42
10.7 Channels 44
10.8 Providers 46

11 Indices and tables 49

HTTP Routing Table 51

CHAPTER 1

Overview

1.1 Format

All response bodies are sent as JSON values.

Where specified, query string parameters can be provided.

Request bodies for `POST`, `PUT`, `PATCH` and `DELETE` requests are expected to be valid JSON values.

1.2 Verbs

Unless otherwise specified, the following verbs are used for requests:

Verb	Description
GET	Used for retrieving a resource
POST	Used for creating a resource
PUT	Used to replace a resource
PATCH	Used to partially update a resource with fields given in the request body
DELETE	Used to delete a resource

1.3 Partial updates

Partial update requests should provide instructions for how a resource should be modified according to the structure proposed in RFC 6902. Note that partial updates are *atomic*, so if any of the given instructions fail, none of the instructures will be carried out.

```
PATCH /projects/23/dialogues/21 HTTP/1.1
Content-Type: application/json-patch+json

[{
  "op": "remove",
  "path": "/sequences/0"
}, {
  "op": "add",
  "path": "/sequences",
  "value": {
    "id": "start",
    "title": "Start of sequence",
    "blocks": []
  }
}]
```

1.4 Client errors

All client error responses contain a `type`, `message` and `details` fields. `type` is a programmitically-usable string representing the type of the error, `message` is a human-readable string describing the error and `details` is an object containing data specific to the error type.

```
HTTP/1.1 400 Bad Request

{
  "type": "parse_error",
  "message": "Invalid JSON in request body",
  "details": {
    "reason": "Expecting value",
    "line": 1,
    "column": 9
  }
}
```

Two client errors common to API endpoints expecting a request body are **parsing errors** and **validation errors**.

1.4.1 Parsing errors

If the data provided in the request body is invalid JSON, a `400 Bad Request` response will be given.

The `details` object contains a human readable `reason` string describing why parsing failed, as well as the `line` and `column` numbers for where parsing failed.

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
  "type": "parse_error",
  "message": "Invalid JSON in request body",
  "details": {
    "reason": "Expecting value",
    "line": 1,
    "column": 9
  }
}
```

1.4.2 Validation errors

If the data provided in the request body is valid JSON, but the data does not fit the schema for that particular request body, a `422 Unprocessable Entity` response will be given.

The `details` object contains an array of `errors` detailing why validation failed. Each error object contains a programmitically-usable `type` string representing the type of validation error, a `path` JSON pointer string pointing to where in the given object validation failed, and a human readable `message` string describing the error.

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: application/json

{
  "type": "validation_error",
  "message": "Invalid request body",
  "details": {
    "errors": [{
      "type", "required",
      "path": "/foo/0/baz",
      "message": "'quux' is a required property"
    }, {
      "type", "type",
      "path": "bar",
      "message": "23 is not of type 'string'"
    }]
  }
}
```

1.5 Read-only fields

Many of the API's *data structures* return read-only fields in API responses. For example, API-generated `id` fields, or a resource's `url` field. If read only fields are provided by clients in API request bodies, the way these are handled depends on the request method.

For `PUT` requests, in order to allow clients to more easily send back an updated description, clients may include read-only fields, but these fields will be ignored when replacing the resource.

For `POST` and `PATCH` requests, if a client provides read-only fields, the API will return an `422 Unprocessable Entity` response, since this case is likely to be a client error:

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: application/json

{
  "type": "validation_error",
  "message": "Invalid request body",
  "details": {
    "errors": [{
      "type", "additionalProperties",
      "path": "/",
      "message": "Additional properties are not allowed ('id' was unexpected)"
    }]
  }
}
```

1.6 Concepts

1.6.1 Users

Users have access to a set of *projects*. Depending on their *permissions*, users can view or modify a project and its *dialogues*.

1.6.2 Projects

A project comprises a set of *dialogues*. End user state is shared across dialogues under the same project.

1.6.3 Dialogues

A dialogue comprises the entire set of steps to follow when a user interacts with a service. It contains a set of *Sequences*, each of which contain a set of *Blocks*.

- *Schema for dialogue descriptions*
- *API endpoints for dialogues*

1.6.4 Sequences

A sequence is a set of steps that follow one after the other, where each step corresponds to a *Block*. A user moves from one sequence to another if different steps need to be followed based on certain conditions.

For example, a dialogue could contain a sequence with a block that asks the user a multiple choice question and moves the user to a different sequence that corresponds to their choice.

1.6.5 Blocks

A block is a single step to follow when interacting with the user. This step may be, for example, a screen asking the user a question, or a step not visible to the user, for example, registering the user with a service.

1.6.6 Symbols

Symbols are used in a *Dialogue* data structure as programmatically-usable strings. Their main use is for identifying and referencing sequences, blocks and block types.

1.6.7 Revision

A revision represents a change or set of changes made to a *dialogue*. There are different revision types:

Edit

Applies a set of changes to a *dialogue description*. For example, changing the content in a *block* or changing the position of a block in a *sequence*.

Revert

Reverts a dialogue's description back to its state at an earlier revision.

1.6.8 Releases

A release marks a point in a dialogue's history of revisions. End users will always interact with the most recently published release's corresponding dialogue description.

1.6.9 Channels

A channel is an address a dialogue can use to interact with end users, for example:

- an sms shortcode (e.g. `2233`)
- a USSD starcode (e.g. `*120*321#`)
- a twitter handle (e.g. `@foo`)

1.6.10 Providers

A provider is a set of *channels* that corresponds to the service providing the channels. For example, Twitter would be the provider for twitter handles.

1.7 Permissions

A user's actions are limited by the permissions they have been granted. Users can be granted the following permissions:

1.7.1 `admin`

Grants create, archive, read and write access for all projects and dialogues, and publish access for all dialogues.

1.7.2 `projects:create`

Grants access to create new projects. Users with this permission obtain `project:admin` access for the projects they create.

1.7.3 `project:admin`

Grants create, archive, read, write and publish access for a given project's dialogues.

1.7.4 `project:dialogues:read`

Grants read access for a given project's dialogues.

1.7.5 `project:dialogues:write`

Grants write access for a given project's dialogues.

1.7.6 `dialogue:read`

Grants read access for a given dialogue.

1.7.7 `dialogue:write`

Grants write access for a given dialogue.

CHAPTER 2

Users

GET /user

Retrieves the *description* of the authenticated user.

```
GET /user HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/1",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson",
  "permissions": [{
    "type": "admin"
  }]
}
```

POST /user/password-changes/

Creates a password change request for the authenticated user using the *details* given in the request body.

```
POST /user/password-changes/ HTTP/1.1
Content-Type: application/json

{
  "old_password": "1337",
  "new_password": "r00t"
}
```

```
HTTP/1.1 204 No Content
Content-Type: application/json
```

GET /users/

Retrieves the *summaries* of all users. Only accessible if the authenticated user has *admin permission*.

```
GET /user HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/1",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson"
}]
```

Query Parameters

- **page** (*number*) – 1-based index of the page of users to show. Defaults to `1`.
- **per_page** (*number*) – Number of users to show per page. Defaults to `30`. Maximum is `100`.

POST /users/

Creates a new user with the *description* given in the request body and returns the created user's *description*, along with the generated `id` field and `url` field for accessing the user description.

```
POST /projects/ HTTP/1.1
Content-Type: application/json

{
  "first_name": "Joan",
  "last_name": "Watson",
  "password": "1337"
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson"
}
```

GET /users/(**str:** *user_id*)

Retrieves the *description* of the user with id `user_id`.

```
GET /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson"
}
```

PUT /users/(**str:** *user_id*)

Replaces the *description* of the user with id `user_id` with the description given in the request body and returns the given description, along with the user's `id` and the `url` for accessing the user's description.

This operation is only accessible to the authenticated user if their user id is `user_id`, or if they have *admin permission*.

```
PUT /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427 HTTP/1.1
Content-Type: application/json

{
  "first_name": "Joan",
  "last_name": "Watson"
}
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson"
}
```

`PATCH /users/` (**str:** *user_id*)

Partially updates the *description* of the user with id `user_id` using the *instructions* given in the request body and returns the given user's description, along with the user's `id` and the `url` for accessing the user's description.

This operation is only accessible to the authenticated user if their user id is `user_id`, or if they have *admin permission*.

```
PATCH /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427 HTTP/1.1
Content-Type: application/json-patch+json

[{
  "op": "replace",
  "path": "/first_name",
  "value": "Joan"
}]
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "url": "/users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427",
  "email": "foo@bar.org",
  "first_name": "Joan",
  "last_name": "Watson"
}
```

`POST /password-resets/`

Creates a *password reset request* for the user with the email address provided in the request body.

If a user with the given email address is found, the user will be sent an email containing a link to be accessed in order to confirm the reset and choose a new password.

```
POST /password-resets/ HTTP/1.1
Content-Type: application/json

{
  "email": "foo@bar.org"
}
```

```
HTTP/1.1 204 No Content
Content-Type: application/json
```

Note: To avoid leaking information on whether a user has a given email address, the API will return a `204` response regardless of whether a user matches the given email address or not.

POST /password-confirmations/

Confirms a password reset using the *confirmation details* given in the request body.

```
POST /password-confirmations/ HTTP/1.1
Content-Type: application/json

{
  "token": "123abc",
  "password": "r00t"
}
```

```
HTTP/1.1 204 No Content
Content-Type: application/json
```

Permissions

GET /users/ (**str:** *user_id*) **/permissions/**

Retrieves the *permissions* accessible to the authenticating user for the user with id `user_id`.

```
GET /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427/permissions/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "9a12594f30220f6a91bde8da961505be",
  "type": "admin"
}]
```

GET /users/ (**str:** *user_id*) **/permissions/**

str: *permission_id* Retrieves the *permission* with id `permission_id` for the user with id `user_id`.

```
GET /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427/permissions/9a12594f30220f6a91bde8da961505be HTT
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "9a12594f30220f6a91bde8da961505be",
  "type": "admin"
}
```

POST /users/ (**str:** *user_id*) **/permissions/**

Creates a new *permission* for the user with id `user_id` and returns the created permission.

```
POST /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427/permissions/ HTTP/1.1
Content-Type: application/json

{
  "type": "projects:admin",
  "details": {
    "project_id": "23"
  }
}
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "2294e0854d66b461eceddbf239f80f04",
  "type": "projects:admin",
  "details": {
    "project_id": "23"
  }
}
```

DELETE /users/(**str:** *user_id*)**/permissions/**

str: *permission_id* Revokes the permission with id `permission_id` for the user with id `user_id` and returns the revoked permission.

```
DELETE /users/0a2d19e0-bb10-4b84-98cc-52a82b6ed427/permissions/2294e0854d66b461eceddbf239f80f04
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "2294e0854d66b461eceddbf239f80f04",
  "type": "projects:admin",
  "details": {
    "project_id": "23"
  }
}
```

CHAPTER 4

Projects

GET /projects/

Retrieves a *summary* of projects the authenticated user has access to.

```
GET /projects/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "1",
  "title": "Maternal Health ZA",
  "url": "/projects/1"
}, {
  "id": "2",
  "title": "Maternal Health MX",
  "url": "/projects/2",
}]
```

Query Parameters

- **page** (*number*) – 1-based index of the page of projects to show. Defaults to `1`.
- **per_page** (*number*) – Number of projects to show per page. Defaults to `30`. Maximum is `100`.

GET /projects/ (**str:** *project_id*)

Retrieves the *description* of the project with id `project_id`.

```
GET /projects/23 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "title": "Maternal Health ZA"
  "url": "/projects/23",
  "is_archived": false,
  "dialogues": [{
    "id": "21",
    "title": "Service Rating Survey",
    "url": "/projects/23/dialogues/21",
    "is_archived": false,
    "is_published": false,
    "has_changes": false,
    "can_view": true,
    "can_edit": true
  }]
}
```

If the project isn't found, a `404` response will be given. The response body's `details` object contains the `id` of the project given in the request.

```
HTTP/1.1 404 Not Found
Content-Type: application/json

{
  "type": "not_found",
  "message": "Project 23 not found",
  "details": {"id": "23"}
}
```

POST /projects/

Creates a new project with the *project description* given in the request body and returns the created projects's description, along with the generated dialogue `id` field and `url` field for accessing the project description.

The authenticated user creating the project is given *project admin* access for the newly created project.

```
POST /projects/ HTTP/1.1
Content-Type: application/json

{
  "title": "Maternal Health ZA",
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "id": "23",
  "url": "/projects/23",
  "title": "Maternal Health ZA",
  "dialogues": [],
  "is_archived": false
}
```

PUT /projects/ (**str:** *project_id*)

Replaces the *description* of the project with id `project_id` with the description given in the request body and returns the given description, along with the projects's `id` and the `url` for accessing the projects's description.

```
PUT /projects/23 HTTP/1.1
Content-Type: application/json

{
  "id": "23",
  "title": "Maternal Health ZA"
}
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "title": "Maternal Health ZA"
  "url": "/projects/23",
  "is_archived": false,
  "dialogues": [{
    "id": "21",
    "title": "Service Rating Survey",
    "url": "/projects/23/dialogues/21",
    "is_archived": false,
    "is_published": false,
    "has_changes": false,
    "can_view": true,
    "can_edit": true
  }]
}
```

PATCH /projects/(**str:** *project_id*)**/**

Partially updates the *description* of the project with id `project_id` with the *instructions* in the request body and returns the given description, along with the projects's `id` and the `url` for accessing the project's description.

```
PATCH /projects/23 HTTP/1.1
Content-Type: application/json-patch+json

[{
  "op": "replace",
  "path": "/title",
  "value": "Maternal Health ZA"
}]
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "title": "Maternal Health ZA"
  "url": "/projects/23",
  "is_archived": false,
  "dialogues": [{
    "id": "21",
    "title": "Service Rating Survey",
    "url": "/projects/23/dialogues/21",
    "is_archived": false,
    "is_published": false,
    "has_changes": false,
    "can_view": true,
    "can_edit": true
  }]
}
```

GET /projects/(**str:** *project_id*)**/channels/**

Retrieves the *descriptions* of the channels accessible to the project with the id `project_id`.

Only accessible if the authenticated user has *admin permission* or has permissions associated with project `project_id`.

```
    GET /projects/21/channels/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "23",
  "url": "/channels/23",
  "project_id": "21",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true,
  "provider": {
    "id": "21",
    "title": "Twitter"
  }
}]
```

CHAPTER 5

Dialogues

GET /projects/ (**str:** *project_id*) **/dialogues/**

Retrieves a *summary* of every dialogue contained in the project with id `project_id`.

```
GET /projects/23/dialogues/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "21",
  "title": "Service Rating Survey",
  "url": "/projects/23/dialogues/21",
  "is_archived": false,
  "is_published": false,
  "has_changes": false,
  "can_view": true,
  "can_edit": true
}]
```

Query Parameters

- **is_archived** (*boolean*) – Filter on whether this dialogue has been archived.
- **is_published** (*boolean*) – Filter on whether this dialogue has been published.
- **has_changes** (*boolean*) – Filter on whether this dialogue has unpublished changes or not.
- **can_view** (*boolean*) – Filter on whether the user can view this dialogue or not.
- **can_edit** (*boolean*) – Filter on whether the user can edit this dialogue or not.

GET /projects/ (**str:** *project_id*) **/dialogues/**

str: *dialogue_id* Retrieves the *description* of the dialogue with id `dialogue_id` in the project with id `project_id`.

```
GET /projecs/23/dialogues/21 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "21",
  "title": "Service Rating Survey",
  "sequences": [],
  "is_archived": false,
  "is_published": false,
  "has_changes": false,
  "can_view": true,
  "can_edit": true
}
```

If the dialogue isn't found, a `404` response will be given. The response body's `details` object contains the `id` given in the request.

```
HTTP/1.1 404 Not Found
Content-Type: application/json

{
  "type": "not_found",
  "message": "Dialogue 21 not found",
  "details": {"id": "21"}
}
```

POST /projects/(**str:** *project_id*)**/dialogues/**

Creates a new dialogue under the project with the id `project_id` using the *dialogue description* given in the request body and returns the created dialogue's description, along with the generated dialogue `id` field and `url` field for accessing the dialogue description.

```
POST /projects/23/dialogues/ HTTP/1.1
Content-Type: application/json

{
  "title": "Service Rating Survey",
  "sequences": []
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "id": "21",
  "url": "/projects/23/dialogues/21",
  "title": "Service Rating Survey",
  "sequences": [],
  "is_archived": false,
  "is_published": false,
  "has_changes": false,
  "can_view": true,
  "can_edit": true
}
```

PUT /projects/(**str:** *project_id*)**/dialogues/**

str: *dialogue_id* Replaces the *description* of the dialogue with id `dialogue_id` in the project with id `project_id` with the description given in the request body and returns the given description, along with the dialogue's `id` and the `url` for accessing the dialogue's description.

Replacing the dialogue creates a new *edit revision* with a JSON patch representing the instructions needed to change the current dialogue description to the new description given in the request body.

```
POST /projects/23/dialogues/21 HTTP/1.1
Content-Type: application/json
```

```
    {
      "title": "Service Rating Survey",
      "sequences": [],
      "is_archived": false
    }
```

```
    HTTP/1.1 200 OK
    Content-Type: application/json

    {
      "id": "21",
      "url": "/projects/23/dialogues/21",
      "title": "Service Rating Survey",
      "sequences": [],
      "is_archived": false,
      "is_published": false,
      "has_changes": false,
       "can_view": true,
       "can_edit": true
    }
```

Warning: If the id of a *sequence* or *block* is changed, the API will regard the changed sequence or block as a new entity. This means all state previously associated to the sequence or block (for example, metrics and translations) will no longer be associated with it.

PATCH /projects/(**str:** *project_id*)**/dialogues/**

str: *dialogue_id* Partially updates the *description* of the dialogue with id `dialogue_id` in the project with id `project_id` with the *instructions* in the request body and returns the given description, along with the dialogue's `id` and the `url` for accessing the dialogue's description.

Partially updating the dialogue creates a new *edit revision* using the provided patch instructions.

```
PATCH /projects/23/dialogues/21 HTTP/1.1
Content-Type: application/json-patch+json

[{
  "op": "remove",
  "path": "/sequences/0"
}, {
  "op": "add",
  "path": "/sequences",
  "value": {
    "id": "start",
    "title": "Start of sequence",
    "blocks": []
  }
}]
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "21",
  "url": "/projects/23/dialogues/21",
  "title": "Service Rating Survey",
  "sequences": [{
    "id": "start",
    "title": "Start of sequence",
    "blocks": []
  }],
  "is_archived": false,
  "is_published": false,
  "has_changes": false,
  "can_view": true,
  "can_edit": true
}
```

Warning: If the `id` of a *sequence* or *block* is changed, the API will regard the changed sequence or block as a new entity. This means all state previously associated to the sequence or block (for example, metrics and translations) will no longer be associated with it.

5.1 Archiving dialogues

A dialogue can be archived by setting `is_archived` to `true` when *replacing* or *partially updating* the dialogue description. The dialogue is still accessible via the api, but will no longer be triggered by any events associated to it.

CHAPTER 6

Revisions

GET /projects/(**str:** *project_id*)**/dialogues/**
str: *dialogue_id***/revisions/** Retrieves the *revisions* of dialogue `dialogue_id` in the project `project_id`.

```
GET /projects/23/dialogues/21/revisions/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "1",
  "number": 1,
  "user_id": "17",
  "created": 1459943775033,
  "type": "edit",
  "details": {
    "id": "start",
    "title": "Start of sequence",
  },
  "properties": {
    "edit_type": "new_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences",
      "value": {
        "id": "start",
        "title": "Start of sequence",
        "blocks": []
      }
    }]
  }
}]
```

Query Parameters

- **page** (*number*) – 1-based index of the page of revisions to show. Defaults to `1`.
- **per_page** (*number*) – Number of revisions to show per page. Defaults to `30`. Maximum is `100`.
- **ordering** (*string*) – The ordering of the returned revisions. If multiple `ordering` parameters are provided, the returned revisions will be sorted by each provided parameter, in the order the parameters were specified. Defaults to `-number`.

6.1 Ordering revisions

Parameter	Description
`number`	Return revisions in ascending order of revision number
`-number`	Return revisions in descending order of revision number
`created`	Return revisions in ascending order of creation date
`-created`	Return revisions in descending order of creation date

POST /projects/(**str:** *project_id*)**/dialogues/**

str: *dialogue_id***/revisions/** Creates a new revision for dialogue `dialogue_id` under project `project_id` using the *description* given in the request body and returns the created revisions's description, along with the generated `id` field and `url` field for accessing the revision description.

Creating a new revision updates the dialogue's description. Any new requests to *retrieve* the dialogue will return a dialogue *description* with the new revision applied.

```
POST /projects/23/dialogues/21/revisions/ HTTP/1.1
Content-Type: application/json

{
  "type": "edit",
  "details": {
    "id": "start",
    "title": "Start of sequence",
  },
  "properties": {
    "edit_type": "new_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences",
      "value": {
        "id": "start",
        "title": "Start of sequence",
        "blocks": []
      }
    }]
  }
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "id": "1",
  "number": 1,
  "user_id": "17",
  "created": 1459943775033,
  "type": "edit",
  "details": {
    "id": "start",
    "title": "Start of sequence",
  },
  "properties": {
    "edit_type": "new_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences",
      "value": {
        "id": "start",
        "title": "Start of sequence",
        "blocks": []
      }
    }]
  }
}
```

It is also possible to create revisions in bulk by providing an array of revisions. In this case, an array of revision descriptions will be returned:

```
POST /projects/23/dialogues/21/revisions/ HTTP/1.1
Content-Type: application/json

[{
  "type": "edit",
  "details": {
    "id": "start",
    "title": "Start of sequence",
  },
  "properties": {
    "edit_type": "new_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences",
      "value": {
        "id": "start",
        "title": "Start of sequence",
        "blocks": []
      }
    }]
  }
}, {
  "type": "edit",
  "details": {
    "id": "start",
    "old_title": "Start of sequence",
    "new_title": "Start"
  },
  "properties": {
    "edit_type": "rename_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences/title",
      "value": "Start"
    }]
  }
}]
```

```
HTTP/1.1 201 Created
Content-Type: application/json

[{
  "id": "1",
  "number": 1,
  "user_id": "17",
  "created": 1459943775033,
  "type": "edit",
  "details": {
    "id": "start",
    "title": "Start of sequence",
  },
  "properties": {
    "edit_type": "new_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences",
      "value": {
        "id": "start",
        "title": "Start of sequence",
        "blocks": []
      }
    }]
  }
}, {
  "id": "2",
  "number": 2,
  "user_id": "17",
  "created": 1459943775033,
  "type": "edit",
  "details": {
    "id": "start",
    "old_title": "Start of sequence",
    "new_title": "Start"
  },
  "properties": {
    "edit_type": "rename_sequence",
    "patch": [{
      "op": "add",
      "path": "/sequences/title",
      "value": "Start"
    }]
  }
}]
```

Note: Creating revisions in bulk is done atomically. If one of the given revisions cannot be created, none of the given revisions will be created.

CHAPTER 7

Releases

GET /projects/ (**str:** *project_id*) **/dialogues/**
str: *dialogue_id* **/releases/** Retrieves the *descriptions* of the *releases* of dialogue `dialogue_id` in the project `project_id`.

```
GET /projects/23/dialogues/21/releases/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "1",
  "number": 1,
  "url": "/projects/23/dialogues/21/releases/1",
  "revision_id": "7"
}]
```

Query Parameters

- **page** (*number*) – 1-based index of the page of releases to show. Defaults to `1`.
- **per_page** (*number*) – Number of releases to show per page. Defaults to `30`. Maximum is `100`.
- **ordering** (*string*) – The ordering of the returned releases. If multiple `ordering` parameters are provided, the returned releases will be sorted by each provided parameter, in the order the parameters were specified. Defaults to `-number`.

7.1 Ordering releases

Parameter	Description
`number`	Return releases in ascending order of release number
`-number`	Return releases in descending order of release number
`created`	Return releases in ascending order of creation date
`-created`	Return releases in descending order of creation date

GET /projects/ (**str:** *project_id*) **/dialogues/**
str: *dialogue_id* **/releases/str:** *release_id* Retrieves the *description* for the release with id `release_id` for dialogue `dialogue_id` contained in the project with id `project_id`.

```
GET /projects/23/dialogues/21/releases/44 HTTP/1.1
```

```
 HTTP/1.1 200 OK
 Content-Type: application/json

{
  "id": "1",
  "number": 1,
  "url": "/projects/23/dialogues/21/releases/1",
  "revision_id": "7",
  "created": 1460022608855
}
```

If the release isn't found, a `404` response will be given. The response body's `details` object contains the `id` and `dialogue_id` given in the request.

```
HTTP/1.1 404 Not Found
Content-Type: application/json

{
  "type": "not_found",
  "message": "Release 44 not found",
  "details": {
    "id": "44",
    "dialogue_id": "21"
  }
}
```

POST /projects/(**str:** *project_id*)**/dialogues/**
str: *dialogue_id***/releases/** Creates a new release for dialogue `dialogue_id` under the project with the id `project_id` using the *description* given in the request body and returns the created releases's description, along with the generated release `id` field and `url` field for accessing the release description.

```
POST /projects/23/dialogues/21/releases/ HTTP/1.1
Content-Type: application/json

{
  "revision_id": "7"
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "id": "1",
  "number": 1,
  "revision_id": "7",
  "created": 1460022608855,
  "url": "/projects/23/dialogues/21/releases/1"
}
```

CHAPTER 8

Channels

GET /channels/

Retrieves the *descriptions* of all channels. Only accessible if the authenticated user has *admin permission*.

```
GET /channels/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "23",
  "url": "/channels/23",
  "project_id": "21",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true,
  "provider": {
    "id": "21",
    "title": "Twitter"
  }
}]
```

GET /channels/(**str:** *channel_id*)

Retrieves the *description* of the channel with the id `channel_id`.

Only accessible if the authenticated user has *admin permission* or has access to a project using the channel `channel_id`.

```
GET /channels/23 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "url": "/channels/23",
  "project_id": "21",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true,
  "provider": {
    "id": "21",
    "title": "Twitter"
  }
}
```

PUT /channels/(**str:** *channel_id*)

Replaces the *description* of the channel with id `channel_id` with the description given in the request body and returns the given description, along with the channel's `id` and the `url` for accessing the channel's description.

This operation is only accessible to the authenticated user if they have *admin permission*.

```
PUT /channels/23 HTTP/1.1
Content-Type: application/json

{
  "id": "23",
  "project_id": "17",
  "url": "/channels/23",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true
}
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "project_id": "17",
  "url": "/channels/23",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true
}
```

PATCH /channels/(**str:** *channel_id*)

Partially updates the *description* of the channel with id `channel_id` using the *instructions* given in the request body and returns the given channel's description, along with the channel's `id` and the `url` for accessing the channel's description.

This operation is only accessible to the authenticated user if they have *admin permission*.

```
PATCH /channels/23 HTTP/1.1
Content-Type: application/json-patch+json

[{
  "op": "replace",
  "path": "/project_id",
  "value": "17"
}]
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "23",
  "project_id": "17",
  "url": "/channels/23",
  "title": "@foo",
  "address": "@foo",
  "type": "twitter",
  "is_available": true
}
```

8.1 Channel access

A channel can only be accessible by one project at a time. A channel can be made accessible to a project by setting `project_id` to the project's id when *replacing* or *partially updating* the channel description.

CHAPTER 9

Providers

GET /providers/

Retrieves the *summaries* of all providers.

Only accessible if the authenticated user has *admin permission*.

```
GET /providers/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
  "id": "21",
  "url": "/providers/21",
  "title": "Twitter"
}, {
  "id": "22",
  "url": "/providers/22",
  "title": "MTN Nigeria"
}]
```

GET /providers/(**str:** *provider_id*)

Retrieves the *description* of the provider with the id `provider_id`.

Only accessible if the authenticated user has *admin permission* or has access to a project using a channel associated with `provider_id`.

```
GET /providers/21 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "21",
  "url": "/providers/21",
  "title": "Twitter"
  "channels": [{
    "id": "23",
    "url": "/channels/23",
    "project_id": 21,
    "title": "@foo",
    "address": "@foo",
    "type": "twitter",
    "is_available": true
```

```
        } ]
    }
```

CHAPTER 10

Data Structures

10.1 Users

10.1.1 User

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The user's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the user description.

  email:
    type: string
    format: email
    readOnly: true
    description: |
      The email address used when authenticating the user.

  first_name:
    type: string
    description: |
      The user's first name.

  last_name:
    type: string
    description: |
      The user's last name.

additionalProperties: false
```

10.1.2 User Summary

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The user's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the user description.

  email:
    type: string
    format: email
    readOnly: true
    description: |
      The email address used when authenticating the user.

  first_name:
    type: string
    readOnly: true
    description: |
      The user's first name.

  last_name:
    type: string
    readOnly: true
    description: |
      The user's last name.

additionalProperties: false
```

10.1.3 New User

```
$ref: '#/definitions/user/user'

properties:
  password:
    type: string
    description: |
      The new user's password.
```

10.1.4 Password change

```
old_password:
  type: 'string'
  description: |
    The user's old password.

new_password:
  type: 'string'
  description: |
    The user's new password.
```

10.1.5 Password reset

```
email:
  type: string
  format: email
  description: |
    The email address for the user who's password to reset.
```

10.1.6 Password reset confirmations

```
token:
  type: 'string'
  description: |
    Token emailed to the user for the password reset.

password:
  type: 'string'
  description: |
    The user's new password.
```

10.2 Projects

10.2.1 Project

```
required:
  - title

properties:
  id:
    type: string
    readOnly: true
    description: |
      The projects's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the project description.

  title:
    type: string
    description: |
      The human-readable title of the project

  is_archived:
    type: boolean
    default: false
    description: |
      Flag representing whether this project has been archived.

  dialogues:
    type: array
    readOnly: true
    items:
      $ref: '#/definitions/dialogue/summary'
    description: |
      A summary of each dialogue contained in this project.

additionalProperties: false
```

10.2.2 Project Summary

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The projects's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the project description.

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the project

  is_archived:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this project has been archived.

additionalProperties: false
```

10.3 Dialogues

10.3.1 Dialogue

```
required:
  - title
  - sequences

properties:
  id:
    type: string
    readOnly: true
    description: |
      The dialogue's identifier (generated by the API)

  revision_id:
    type: string
    readOnly: true
    description: |
      Identifier of the revision associated with the dialogue description.

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the dialogue description.

  title:
    type: string
    description: |
      The human-readable title of the dialogue

  sequences:
    type: array
    items:
      $ref: '#/definitions/dialogue/sequence'
    description: |
      The dialogue's set of sequences

  is_archived:
    type: boolean
    default: false
    description: |
      Flag representing whether this dialogue has been archived.

  is_published:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this dialogue has been published before.

  has_changes:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this dialogue has unpublished changes.

  can_view:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether the authenticated user can view this dialogue.

  can_edit:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether the authenticated user can edit this dialogue.

additionalProperties: false
```

10.3.2 Dialogue Summary

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The dialogue's identifier (generated by the API)

  revision_id:
    type: string
    readOnly: true
    description: |
      Identifier of the revision associated with the dialogue description.

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the dialogue description.

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the dialogue

  is_archived:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this dialogue has been archived.

  is_published:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this dialogue has been published before.

  has_changes:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether this dialogue has unpublished changes.

  can_view:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether the authenticated user can view this dialogue.

  can_edit:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether the authenticated user can edit this dialogue.

additionalProperties: false
```

10.3.3 Sequence

```
required:
  - id
  - title
  - blocks

properties:
  id:
    $ref: '#/definitions/dialogue/symbol'
    description: |
      The sequence's identifier. Must be generated by the client.

  title:
    type: text
    description: |
      A human-readable title describing this sequence.

  blocks:
    type: array
    items:
      $ref: '#/definitions/dialogue/block'
    description: |
      The sequence's set of blocks

additionalProperties: false
```

10.3.4 Block

Note: A block's `properties` object is validated against a schema corresponding to the block's type (represented by the `type` field).

```
required:
  - id
  - type

properties:
  id:
    $ref: '#/definitions/dialogue/symbol'
    description: |
      The block's identifier. Must be generated by the client.

  type:
    $ref: '#/definitions/dialogue/symbol'
    description: |
      Identifies the type of block. Valid values depend on which block types
      have been registered with a particular instance of the API.

  title:
    type: text
    description: |
      A human-readable title describing the block.

  properties:
    type: object
    description: |
      Properties relevant to the block's type. For example, a block that asks
      the user a question may have a `question` property here.

additionalProperties: false
```

10.3.5 Symbol

```
type: string
pattern: '^[a-z][a-z0-9-]*$'
```

10.4 Revisions

10.4.1 Revision

```
required:
  - patch

properties:
  id:
    type: string
    readOnly: true
    description: |
      The revisions's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the revision description.

  user_id:
    type: string
    readOnly: true
    description: |
      The identifier for the user who created this revision.

  number:
    type: number
    readOnly: true
    description: |
      The revision's number, starting at `1` for the first revision, and
      incremented for each new revision.

  created:
    type: number
    description: |
      Timestamp representing when the revision was created in milliseconds
      since the Unix epoch. Generated by the API if not provided.

  type:
    type: string
    enum:
      - edit
      - revert
    description: |
      Programmatically usable string representing the type of this revision.

  properties:
    type: object
    description: |
      Properties specific to this revision type.

  details:
    type: object
    default: {}
    description: |
      Details describing this revision. Intended to be used for
      display purposes.
```

Edit

```
$ref: '#/definitions/revision/revision'

properties:
  properties:
    edit_type:
      type: string
      description: |
        Programmatically usable string representing the type of edit done in
        this revision.

    patch:
      $ref: 'http://json.schemastore.org/json-patch'
      description: |
        JSON patch representing the changes made in this revision
```

Revert

```
$ref: '#/definitions/revision/revision'

properties:
  properties:
    revision_id:
      type: string
      description: |
        Identifier of the revision to revert to.
```

10.5 Releases

10.5.1 Release

```
required:
  - revision_id
```

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The releases's identifier (generated by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the release's description.

  number:
    type: number
    readOnly: true
    description: |
      The releases's number, starting at `1` for the first release, and
      incremented for each new release.

  created:
    type: number
    readOnly: true
    description: |
      Timestamp representing when the revision was created in milliseconds
      since the Unix epoch.

  revision_id:
    type: string
    description: |
      Identifier of the revision this release is for.

additionalProperties: false
```

10.6 Permissions

```
required:
  - type

properties:
  id:
    type: string
    readOnly: true
    description: |
      The permission's identifier (generated by the API)

  type:
    type: string
    enum:
      - admin
      - projects:create
      - project:admin
      - project:dialogues:read
      - project:dialogues:write
      - dialogue:read
      - dialogue:write
    description: |
      Identifies the type of this permission.

  properties:
    type: object
    default: {}
    description: |
      Properties relevant to the permission type.
```

10.6.1 project:admin

```
$ref: '#/definitions/permission/permission'

properties:
  properties:
    type: object
    properties:
      project_id:
        type: string
        description: |
          Identifier of the project this permission is for.
```

10.6.2 project:dialogues:read

```
$ref: '#/definitions/permission/permission'

properties:
  properties:
    type: object
    properties:
      project_id:
        type: string
        description: |
          Identifier of the project this permission is for.
```

10.6.3 project:dialogues:write

```
$ref: '#/definitions/permission/permission'

properties:
  properties:
    type: object
    properties:
      project_id:
        type: string
        description: |
          Identifier of the project this permission is for.
```

10.6.4 `dialogue:read`

```
$ref: '#/definitions/permission/permission'

properties:
  properties:
    type: object
    properties:
      project_id:
        type: string
        description: |
          Identifier of the project this permission is for.

      dialogue_id:
        type: string
        description: |
          Identifier of the dialogue this permission is for.
```

10.6.5 `dialogue:write`

```
$ref: '#/definitions/permission/permission'

properties:
  properties:
    type: object
    properties:
      project_id:
        type: string
        description: |
          Identifier of the project this permission is for.

      dialogue_id:
        type: string
        description: |
          Identifier of the dialogue this permission is for.
```

10.7 Channels

10.7.1 Channel

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The channel's identifier (provided by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the channel description
  project_id:
    type:
      - string
      - 'null'
    description: |
      Identifier for the project currently using this channel. `null` if the
      channel is not in use by any projects.

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the channel

  type:
    type: string
    readOnly: true
    description: |
      Programmatically usable string representing the type of this channel. For
      e.g. `sms`, `ussd`.

  address:
    type: string
    readOnly: true
    description: |
      The address associated with this channel. For example, ``*120*321#``.

  is_available:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether or not this channel is available for use.

  provider:
    $ref: '#/definitions/provider/summary'
    readOnly: true
    description: |
      Summary of the provider that associated with this channel.

additionalProperties: false
```

10.7.2 Channel Summary

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The channel's identifier (provided by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the channel description
  project_id:
    readOnly: true
    type:
      - string
      - 'null'
    description: |
      Identifier for the project currently using this channel. `null` if the
      channel is not in use by any projects.

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the channel

  type:
    type: string
    readOnly: true
    description: |
      Programmatically usable string representing the type of this channel. For
      e.g. `sms`, `ussd`.

  address:
    type: string
    readOnly: true
    description: |
      The address associated with this channel. For example, ``*120*321#``.

  is_available:
    type: boolean
    readOnly: true
    description: |
      Flag representing whether or not this channel is available for use.

additionalProperties: false
```

10.8 Providers

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The channel's identifier (provided by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the provider description

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the provider
  channels:
    type: array
    readOnly: true
    items:
      $ref: '#/definitions/channel/summary'
    description: |
      Summaries of the channels associated with this provider

additionalProperties: false
```

10.8.1 Provider Summary

```
properties:
  id:
    type: string
    readOnly: true
    description: |
      The channel's identifier (provided by the API)

  url:
    type: string
    readOnly: true
    description: |
      API url for accessing the provider description

  title:
    type: string
    readOnly: true
    description: |
      The human-readable title of the provider

additionalProperties: false
```

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

/channels

GET /channels/, 27
GET /channels/(str:channel_id), 27
PUT /channels/(str:channel_id), 28
PATCH /channels/(str:channel_id), 28

/password-confirmations

POST /password-confirmations/, 10

/password-resets

POST /password-resets/, 9

/projects

GET /projects/, 13
GET /projects/(str:project_id), 13
GET /projects/(str:project_id)/channels/, 15
GET /projects/(str:project_id)/dialogues/, 17
GET /projects/(str:project_id)/dialogues/(str:dialogue_id), 17
GET /projects/(str:project_id)/dialogues/(str:dialogue_id)/releases/, 25
GET /projects/(str:project_id)/dialogues/(str:dialogue_id)/releases/(str:release_id), 25
GET /projects/(str:project_id)/dialogues/(str:dialogue_id)/revisions/, 21
POST /projects/, 14
POST /projects/(str:project_id)/dialogues/, 18
POST /projects/(str:project_id)/dialogues/(str:dialogue_id)/releases/, 26
POST /projects/(str:project_id)/dialogues/(str:dialogue_id)/revisions/, 22
PUT /projects/(str:project_id), 14
PUT /projects/(str:project_id)/dialogues/(str:dialogue_id), 18
PATCH /projects/(str:project_id)/, 15
PATCH /projects/(str:project_id)/dialogues/(str:dialogue_id), 19

/providers

GET /providers/, 31
GET /providers/(str:provider_id), 31

/user

GET /user, 7
POST /user/password-changes/, 7

/users

GET /users/, 7
GET /users/(str:user_id), 8
GET /users/(str:user_id)/permissions/, 11
GET /users/(str:user_id)/permissions/(str:permissio 11
POST /users/, 8
POST /users/(str:user_id)/permissions/, 11
PUT /users/(str:user_id), 8
DELETE /users/(str:user_id)/permissions/(str:permis 12
PATCH /users/(str:user_id), 9