Reading List Documentation

Release 2.0.0

Mozilla Services — Da French Team

June 28, 2016

Contents

Reading List is a service that aims to synchronize a list of articles URLs between a set of devices owned by a same account.

- Online documentation
- Issue tracker
- Contributing

CHAPTER 1

Table of content

1.1 API Endpoints

1.1.1 Authentication

Firefox Account OAuth Bearer token

Use the OAuth token with this header:

```
Authorization: Bearer <oauth_token>
```

Obtain the token

Using the Web UI

- Navigate the client to `GET /v2/fxa-oauth/login?redirect=http://app-endpoint/#`. There, a session cookie will be set, and the client will be redirected to a login form on the FxA content server
- After submitting the credentials on the login page, the client will be redirected to `http://app-endpoint/#{token}` the web-app.

Custom flow The `GET /v2/fxa-oauth/params` endpoint can be use to get the configuration in order to trade the Firefox Account BrowserID with a Bearer Token. See Firefox Account documentation about this behavior

```
$ http GET http://localhost:8000/v2/fxa-oauth/params -v

GET /v2/fxa-oauth/params HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Host: localhost:8000
User-Agent: HTTPie/0.8.0


HTTP/1.1 200 OK
Content-Length: 103
Content-Type: application/json; charset=UTF-8
Date: Thu, 19 Feb 2015 09:28:37 GMT
Server: waitress

{
```

```
    "client_id": "89513028159972bc",
    "oauth_uri": "https://oauth-stable.dev.lcip.org",
    "scope": "readinglist"
}
```

Basic Auth

In addition to OAuth, *Basic Auth* can be enabled in the configuration using `cliquet.basic_auth_enabled = true`.

Articles will then be stored for any username/password combination provided.

1.1.2 Resource endpoints

In this section, the request example provided are performed using httpie .

GET /articles

Requires authentication

Returns all records of the current user for this resource.

The returned value is a JSON mapping containing:

- `items`: the list of records, with exhaustive attributes

A `Total-Records` header is sent back to indicate the estimated total number of records included in the response.

A header `Last-Modified` will provide the current timestamp of the collection (*see Server timestamps section*). It is likely to be used by client to provide `If-Modified-Since` or `If-Unmodified-Since` headers in subsequent requests.

Filtering

Single value

- `/articles?unread=true`

Minimum and maximum

Prefix attribute name with `min_` or `max_`:

- `/articles?min_word_count=4000`

  note The lower and upper bounds are inclusive (*i.e equivalent to greater or equal*).

  note `lt_` and `gt_` can also be used to exclude the bound.

Exclude

Prefix attribute name with `not_`:

- `/articles?not_read_position=0`

  note Will return an error if a field is unknown.

  note The `Last-Modified` response header will always be the same as the unfiltered collection.

Sorting

- `/articles?_sort=-last_modified,title`

  note Ordering on a boolean field gives `true` values first.

  note Will return an error if a field is unknown.

Counting

In order to count the number of records, for a specific field value for example, without fetching the actual collection, a `HEAD` request can be used. The `Total-Records` response header will then provide the total number of records.

See *batch endpoint* to count several collections in one request.

Polling for changes

The `_since` parameter is provided as an alias for `gt_last_modified`.

- `/articles?_since=123456`

The new value of the collection latest modification is provided in headers (*see Server timestamps section*).

When filtering on `last_modified` (i.e. with `_since` or `_to` parameters), every deleted articles will appear in the list with a deleted status (`deleted=true`).

If the request header `If-Modified-Since` is provided, and if the collection has not suffered changes meanwhile, a `304 Not Modified` response is returned.

note The `_to` parameter is also available, and is an alias for `lt_last_modified` (*strictly inferior*).

Paginate

If the `_limit` parameter is provided, the number of items is limited.

If there are more items for this collection than the limit, the response will provide a `Next-Page` header with the URL for the Next-Page.

When there is not more `Next-Page` response header, there is nothing more to fetch.

Pagination works with sorting and filtering.

List of available URL parameters

- `<prefix?><attribute name>`: filter by value(s)
- `_since`: polling changes
- `_sort`: order list
- `_limit`: pagination max size
- `_token`: pagination token

Combining all parameters

Filtering, sorting and paginating can all be combined together.

- `/articles?_sort=-last_modified&_limit=100`

Example

```
http POST http://localhost:8000/v2/articles?_sort=-last_modified -v --auth "admin:"
```

```
GET /v2/articles?_sort=-last_modified HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Host: localhost:8000
User-Agent: HTTPie/0.8.0

HTTP/1.1 200 OK
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Total-Records, Alert, Next-Page
Content-Length: 610
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:20:08 GMT
Last-Modified: 1425053903124
Server: waitress
Total-Records: 1

{
    "items": [
        {
            "added_by": "Natim",
            "added_on": 1425053903123,
            "excerpt": "",
            "favorite": false,
            "id": "ff795c43c02145a4b7a5df5260ee182d",
            "is_article": true,
            "last_modified": 1425053903124,
            "marked_read_by": null,
            "marked_read_on": null,
            "read_position": 0,
            "resolved_title": "The Hawk Authorization protocol",
            "resolved_url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-i
            "archived": false,
            "stored_on": 1425053903123,
            "title": "The Hawk Authorization protocol",
            "unread": true,
            "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
            "word_count": null
        }
    ]
}
```

POST /articles

Requires authentication

Used to create a record on the server. The POST body is a JSON mapping containing the values of the resource schema fields.

- `url`
- `title`
- `added_by`

The POST response body is the newly created record, if all posted values are valid.

If the request header `If-Unmodified-Since` is provided, and if the record has changed meanwhile, a `412 Precondition failed` error is returned.

To get a list of mandatory, optional and default values, refer to the *Data model*.

Validation

If the posted values are invalid (e.g. *field value is not an integer*) an error response is returned with status `400`.

Conflicts

The Reading List Server provides an automatic conflict resolution algorithm for articles.

An article is uniquely defined in the database by its `url` and `resolved_url` fields. The `resolved_url` field can be different from the `url` value when the client has to follow one or several redirections.

note Unicity on URLs is determined by the full URL, including location hash. (e.g. http://news.com/day-1.html#paragraph1, http://spa.com/#/content/3)

note Deleted records are not taken into account for field unicity.

When a client pushes a new article which `url` or `resolved_url` already exists in the database, the automatic conflict resolver will simply keep the original one with all its values (title, summary etc.) and return to the client its information. No duplicate is created.

Updating the title, excerpt or word_count of an existing article won't raise any conflict: the last call wins.

For both updates and creation, the automatic conflict resolution can be bypassed with a `If-Unmodified-Since` in requests headers.

In that case, a `412 Precondition failed` error is returned: - for a `POST` on the collection if something was changed in the database

in the interim.

- for a `PATCH` on an article that was changed in the interim.

note If two articles with conflicting URLs are posted in the same *batch*, the same behavior as described above will apply, as the batch queries are processed in a sequential order.

Example

```
http POST http://localhost:8000/v2/articles \
    title="The Hawk Authorization protocol" \
    url=https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/ \
    added_by=Natim \
    --auth "admin:" -v
```

```
POST /v2/articles HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Content-Length: 150
Content-Type: application/json; charset=utf-8
Host: localhost:8000
User-Agent: HTTPie/0.8.0

{
    "added_by": "Natim",
    "title": "The Hawk Authorization protocol",
    "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/"
}

HTTP/1.1 201 Created
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Total-Records, Alert, Next-Page
Content-Length: 597
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:18:23 GMT
Server: waitress

{
    "added_by": "Natim",
    "added_on": 1425053903123,
    "archived": false,
    "excerpt": "",
    "favorite": false,
    "id": "ff795c43c02145a4b7a5df5260ee182d",
    "is_article": true,
    "last_modified": 1425053903124,
    "marked_read_by": null,
    "marked_read_on": null,
    "read_position": 0,
    "resolved_title": "The Hawk Authorization protocol",
    "resolved_url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "stored_on": 1425053903123,
    "title": "The Hawk Authorization protocol",
    "unread": true,
    "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "word_count": null
}
```

DELETE /articles

Requires authentication

Delete multiple records. **Enabled by default**, see recommended production settings to disable.

The DELETE response is a JSON mapping with an `items` attribute, returning the list of records that were deleted.

It supports the same filtering capabilities as GET.

If the request header `If-Unmodified-Since` is provided, and if the collection has changed meanwhile, a `412 Precondition failed` error is returned.

Example

```
http DELETE http://localhost:8000/v2/articles \
    --auth "admin:" -v
```

```
DELETE /v2/articles HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Content-Length: 0
Host: localhost:8000
User-Agent: HTTPie/0.8.0


HTTP/1.1 200 OK
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Alert
Content-Length: 100
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:27:55 GMT
Server: waitress

{
    "items": [
        {
            "deleted": true,
            "id": "30afb809ca7745a58496a09c6a4afcac",
            "last_modified": 1425054475110
        }
    ]
}
```

GET /articles/<id>

Requires authentication

Returns a specific record by its id.

For convenience and consistency, a header `Last-Modified` will also repeat the value of `last_modified`.

If the request header `If-Modified-Since` is provided, and if the record has not changed meanwhile, a `304 Not Modified` is returned.

Example

```
http GET http://localhost:8000/v2/articles/30afb809ca7745a58496a09c6a4afcac \
    --auth "admin:" -v
```

```
GET /v2/articles/30afb809ca7745a58496a09c6a4afcac HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Host: localhost:8000
User-Agent: HTTPie/0.8.0


HTTP/1.1 200 OK
```

```
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Alert
Content-Length: 597
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:22:38 GMT
Last-Modified: 1425054146681
Server: waitress

{
    "added_by": "Natim",
    "added_on": 1425054146680,
    "archived": false,
    "excerpt": "",
    "favorite": false,
    "id": "30afb809ca7745a58496a09c6a4afcac",
    "is_article": true,
    "last_modified": 1425054146681,
    "marked_read_by": null,
    "marked_read_on": null,
    "read_position": 0,
    "resolved_title": "The Hawk Authorization protocol",
    "resolved_url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "stored_on": 1425054146680,
    "title": "The Hawk Authorization protocol",
    "unread": true,
    "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "word_count": null
}
```

DELETE /articles/<id>

Requires authentication

Delete a specific record by its id.

The DELETE response is the record that was deleted.

If the record is missing (or already deleted), a `404 Not Found` is returned. The client might decide to ignore it.

If the request header `If-Unmodified-Since` is provided, and if the record has changed meanwhile, a `412 Precondition failed` error is returned.

> **note** Once deleted, an article will appear in the collection with a deleted status (`deleted=true`) and will have most of its fields empty.

Example

```
http DELETE http://localhost:8000/v2/articles/ff795c43c02145a4b7a5df5260ee182d \
    --auth "admin:" -v
```

```
DELETE /v2/articles/ff795c43c02145a4b7a5df5260ee182d HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Content-Length: 0
Host: localhost:8000
User-Agent: HTTPie/0.8.0
```

```
HTTP/1.1 200 OK
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Alert
Content-Length: 87
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:21:00 GMT
Server: waitress

{
    "deleted": True,
    "id": "ff795c43c02145a4b7a5df5260ee182d",
    "last_modified": 1425054060041
}
```

PATCH /articles/<id>

Requires authentication

Modify a specific record by its id. The PATCH body is a JSON mapping containing a subset of articles fields.

The PATCH response is the modified record (full).

Modifiable fields

- `title`
- `excerpt`
- `favorite`
- `unread`
- `archived`
- `read_position`

Since article fields resolution is performed by the client in the first version of the API, the following fields are also modifiable:

- `is_article`
- `resolved_url`
- `resolved_title`

Response behavior

On a `PATCH` it is possible to choose among different behaviors for the response content.

Three behaviors are available:

- `full`: Returns the whole record (**default**).
- `light`: Returns only the fields whose value was changed.
- `diff`: Returns only the fields values that don't match those provided.

For example, using the default behavior :

```
http PATCH http://localhost:8000/v2/articles/8412b7d7da40467e9afbad8b6f15c20f \
    unread=False marked_read_on=1425316211577 marked_read_by=Ipad \
    --auth 'Natim:' -v
```

```
PATCH /v2/articles/8412b7d7da40467e9afbad8b6f15c20f HTTP/1.1
Host: localhost:8000
[...]

{
    "marked_read_by": "Ipad",
    "marked_read_on": "1425316211577",
    "unread": "False"
}

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
[...]

{
    "added_by": "Natim",
    "added_on": 1425383479321,
    "archived": false,
    "excerpt": "",
    "favorite": false,
    "id": "8412b7d7da40467e9afbad8b6f15c20f",
    "is_article": true,
    "last_modified": 1425383532546,
    "marked_read_by": "Ipad",
    "marked_read_on": 1425316211577,
    "read_position": 0,
    "resolved_title": "What's Hawk authentication and how to use it?",
    "resolved_url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "stored_on": 1425383479321,
    "title": "The Hawk Authorization protocol",
    "unread": false,
    "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "word_count": null
}
```

Using `Response-Behavior: light`

```
http PATCH http://localhost:8000/v2/articles/8412b7d7da40467e9afbad8b6f15c20f \
    unread=False marked_read_on=1425316211577 marked_read_by=Ipad \
    Response-Behavior:light \
    --auth 'Natim:' -v
```

```
PATCH /v2/articles/8412b7d7da40467e9afbad8b6f15c20f HTTP/1.1
Host: localhost:8000
Response-Behavior: light
[...]

{
    "marked_read_by": "Ipad",
    "marked_read_on": "1425316211577",
    "unread": "False"
}

HTTP/1.1 200 OK
```

```
[...]
Content-Type: application/json; charset=UTF-8

{
    "marked_read_by": "Ipad",
    "marked_read_on": 1425316211577,
    "unread": false
}
```

Using `Response-Behavior: diff`

```
http PATCH http://localhost:8000/v2/articles/8412b7d7da40467e9afbad8b6f15c20f \
    unread=False marked_read_on=1425316211577 marked_read_by=Ipad \
    Response-Behavior:diff \
    --auth 'Natim:' -v
```

```
PATCH /v2/articles/8412b7d7da40467e9afbad8b6f15c20f HTTP/1.1
Host: localhost:8000
Response-Behavior: diff
[...]

{
    "marked_read_by": "Ipad",
    "marked_read_on": "1425316211577",
    "unread": "False"
}

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
[...]

{}
```

Errors

If a read-only field is modified, a `400 Bad request` error is returned.

If the record is missing (or already deleted), a `404 Not Found` error is returned. The client might decide to ignore it.

If the request header `If-Unmodified-Since` is provided, and if the record has changed meanwhile, a `412 Precondition failed` error is returned.

note `last_modified` is updated to the current server timestamp, only if a field value was changed.

note Changing `read_position` never generates conflicts.

note `read_position` is ignored if the value is lower than the current one.

note If `unread` is changed to false, `marked_read_on` and `marked_read_by` are expected to be provided.

note If `unread` was already false, `marked_read_on` and `marked_read_by` are not updated with provided values.

note If `unread` is changed to true, `marked_read_by`, `marked_read_on` and `read_position` are reset to their default value.

Conflicts

If changing the article `resolved_url` violates the unicity constraint, a `409 Conflict` error response is returned (see error channel).

note Note that `url` is a readonly field, and thus cannot generate conflicts here.

Example

```
http PATCH http://localhost:8000/v2/articles/30afb809ca7745a58496a09c6a4afcac \
    title="What's Hawk authentication and how to use it?" \
    If-Unmodified-Since:1425054146681 \
    --auth "admin:" -v
```

```
PATCH /v2/articles/30afb809ca7745a58496a09c6a4afcac HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46
Content-Length: 63
Content-Type: application/json; charset=utf-8
Host: localhost:8000
If-Unmodified-Since: 1425054146681
User-Agent: HTTPie/0.8.0

{
    "title": "What's Hawk authentication and how to use it?"
}

HTTP/1.1 200 OK
Access-Control-Expose-Headers: Backoff, Retry-After, Last-Modified, Alert
Content-Length: 616
Content-Type: application/json; charset=UTF-8
Date: Fri, 27 Feb 2015 16:24:21 GMT
Server: waitress

{
    "added_by": "Natim",
    "added_on": 1425054146680,
    "archived": false,
    "excerpt": "",
    "favorite": false,
    "id": "30afb809ca7745a58496a09c6a4afcac",
    "is_article": true,
    "last_modified": 1425054261938,
    "marked_read_by": null,
    "marked_read_on": null,
    "read_position": 0,
    "resolved_title": "The Hawk Authorization protocol",
    "resolved_url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "stored_on": 1425054146680,
    "title": "What's Hawk authentication and how to use it?",
    "unread": true,
    "url": "https://blog.mozilla.org/services/2015/02/05/whats-hawk-and-how-to-use-it/",
    "word_count": null
}
```

1.1.3 Batch operations

POST /batch

Requires an FxA OAuth authentication

The POST body is a mapping, with the following attributes:

- `requests`: the list of requests (*limited to 25 by default*)
- `defaults`: (*optional*) default requests values in common for all requests

  Each request is a JSON mapping, with the following attribute:

- `method`: HTTP verb
- `path`: URI
- `body`: a mapping
- `headers`: (*optional*), otherwise take those of batch request

```
{
  "defaults": {
    "method" : "POST",
    "path" : "/articles",
    "headers" : {
      ...
    }
  },
  "requests": [
    {
      "body" : {
        "title": "MoFo",
        "url" : "http://mozilla.org",
        "added_by": "FxOS",
      }
    },
    {
      "body" : {
        "title": "MoCo",
        "url" : "http://mozilla.com"
        "added_by": "FxOS",
      }
    },
    {
      "method" : "PATCH",
      "path" : "/articles/409",
      "body" : {
        "read_position" : 3477
      }
    }
  ]
]
```

The response body is a list of all responses:

```
{
  "responses": [
    {
      "path" : "/articles/409",
```

```
      "status": 200,
      "body" : {
        "id": 409,
        "url": "...",
        ...
        "read_position" : 3477
      },
      "headers": {
        ...
      }
    },
    {
      "status": 201,
      "path" : "/articles",
      "body" : {
        "id": 411,
        "title": "MoFo",
        "url" : "http://mozilla.org",
        ...
      },
    },
    {
      "status": 201,
      "path" : "/articles",
      "body" : {
        "id": 412,
        "title": "MoCo",
        "url" : "http://mozilla.com",
        ...
      },
    },
  ]
]
```

warning Since the requests bodies are necessarily mappings, posting arbitrary data (*like raw text or binary*)is not supported.

note Responses are provided in the same order than requests.

note A form of payload optimization for massive operations is planned.

1.1.4 Utility endpoints for OPS and Devs

GET /

The returned value is a JSON mapping containing:

- `hello`: the name of the service (e.g. `"reading list"`)
- `version`: complete version (`"X.Y.Z"`)
- `url`: absolute URI (without a trailing slash) of the API (*can be used by client to build URIs*)
- `eos`: date of end of support in ISO 8601 format (`"yyyy-mm-dd"`, undefined if unknown)
- `documentation`: The url to the service documentation. (this document!)

GET /__heartbeat__

Return the status of each service your application depends on. The returned value is a JSON mapping containing:

- `database` true if operational

Return `200` if the connection with each service is working properly and `503` if something doesn't work.

1.1.5 Server timestamps

In order to avoid race conditions, all timestamps manipulated by the server are not true HTTP date values, nor milliseconds EPOCH timestamps.

They are milliseconds EPOCH timestamps with the guarantee of a change per timestamp update. If two changes happen at the same millisecond, they will have two differents timestamps.

The `Last-Modified` header with the last timestamp of the collection for a given user will be given on collection and record GET endpoints.

```
Last-Modified: 1422375916186
```

> **note** Both fields `added_on` and `marked_on` will contain actual timestamps (from device perspective), used for calendar year information display.

All timestamp of the app will be set in milliseconds.

1.1.6 API versioning

Versionning

The API versioning is based on the application version deployed. It follows semver.

During development the server will be 0.X.X, the server endpoint will be prefixed by `/v2`.

Each non retro-compatible API change will imply the major version number to be incremented. Everything will be made to avoid retro incompatible changes.

The `/` endpoint will redirect to the last API version.

Deprecation

A track of the client version will be kept to know after which date each old version can be shutdown.

The date of the end of support is provided in the API root URL (e.g. `/v2`)

Using the `Alert` response header, the server can communicate any potential warning messages, information, or other alerts.

The value is JSON mapping with the following attributes:

- `code`: one of the strings `"soft-eol"` or `"hard-eol"`;
- `message`: a human-readable message (optional);
- `url`: a URL at which more information is available (optional).

A `410 Gone` error response can be returned if the client version is too old, or the service had been remplaced with a new and better service using a new protocol version.

1.1.7 Backoff indicators

Backoff header on heavy load

A `Backoff` header will be added to the success responses (>=200 and <400) when the server is under heavy load. It provides the client with a number of seconds during which it should avoid doing unnecessary requests.

```
Backoff: 30
```

note The back-off time is configurable on the server.

note In other implementations at Mozilla, there was `X-Weave-Backoff` and `X-Backoff` but the `X-` prefix for header has been deprecated since.

Retry-After indicators

A `Retry-After` header will be added to error responses (>=500), telling the client how many seconds it should wait before trying again.

```
Retry-After: 30
```

1.1.8 Error responses

Protocol description

Every response is JSON.

If the HTTP status is not OK (<200 or >=400), the response contains a JSON mapping, with the following attributes:

- `code`: matches the HTTP status code (e.g `400`)
- `errno`: stable application-level error number (e.g. `109`)
- `error`: string description of error type (e.g. `"Bad request"`)
- `message`: context information (e.g. `"Invalid request parameters"`)
- `info`: additional details (e.g. URL to error details)

Example response

```
{
    "code": 400,
    "errno": 109,
    "error": "Bad Request",
    "message": "Invalid posted data",
    "info": "https://server/docs/api.html#errors"
}
```

Error codes

status code	errno	description
401	104	Missing Authorization Token
401	105	Invalid Authorization Token
400	106	request body was not valid JSON
400	107	invalid request parameter
400	108	missing request parameter
400	109	invalid posted data
404	110	Invalid Token / id
404	111	Missing Token / id
411	112	Content-Length header was not provided
413	113	Request body too large
412	114	Resource was modified meanwhile
405	115	Method not allowed on this end point
429	117	Client has sent too many requests
403	121	Resource's access forbidden for this user
409	122	Another resource violates constraint
500	999	Internal Server Error
503	201	Service Temporary unavailable due to high load
410	202	Service deprecated

Validation errors

In case multiple validation errors occur on a request, they will be returned one at a time.

1.2 Data model

1.2.1 Articles

Attribute	Type	Comment	Mandatory*	Default Value
`id`	UUID			uuid()
`last_modified`	Timestamp	server timestamp		
`url`	URL	valid (RFC)	YES	
`preview`	URL	feature image URL		
`title`	String(1024)	1 character min.	YES	
`resolved_url`	URL	after potential redirections		url
`resolved_title`	String(1024)	extracted from target page		title
`excerpt`	Text	first 200 words of the article		empty text
`archived`	Boolean			false
`favorite`	Boolean			false
`is_article`	Boolean	false if no textual content		true
`word_count`	Integer			null
`unread`	Boolean			true
`added_by`	Device	device name (cf. issue #23)	YES	
`added_on`	Timestamp	device timestamp		server time
`stored_on`	Timestamp	server timestamp		server time
`marked_read_by`	Device	device name (cf. issue #23)		null
`marked_read_on`	Timestamp	device timestamp		null
`read_position`	Integer	Words read from the beginning		0

Mandatory : The field has to be provided on creation.

1.3 Installation

1.3.1 Run locally

Reading List is based on top of the cliquet project, and as such, please refer to cliquet's documentation for more details.

For development

By default, *Reading List* persists the records and internal cache in a PostgreSQL database.

The default configuration will connect to the `postgres` database on `localhost:5432`, with user/password `postgres/postgres`. See more details below about installation and setup of PostgreSQL.

```
make serve
```

Using Docker

Reading List uses Docker Compose, which takes care of running and connecting PostgreSQL:

```
docker-compose up
```

Authentication

By default, *Reading List* relies on Firefox Account OAuth2 Bearer tokens to authenticate users.

See cliquet documentation to configure authentication options.

1.3.2 Install and setup PostgreSQL

(*requires PostgreSQL 9.3 or higher*).

Using Docker

```
docker run -e POSTGRES_PASSWORD=postgres -p 5434:5432 postgres
```

Linux

On debian / ubuntu based systems:

```
apt-get install postgresql postgresql-contrib
```

By default, the `postgres` user has no password and can hence only connect if ran by the `postgres` system user. The following command will assign it:

```
sudo -u postgres psql -c "ALTER USER postgres PASSWORD 'postgres';"
```

OS X

Assuming brew is installed:

```
brew update
brew install postgresql
```

Create the initial database:

```
initdb /usr/local/var/postgres
```

1.3.3 Cryptography libraries

Linux

On Debian / Ubuntu based systems:

```
apt-get install libffi-dev libssl-dev
```

On RHEL-derivatives:

```
apt-get install libffi-devel openssl-devel
```

OS X

Assuming brew is installed:

```
brew install libffi openssl pkg-config
```

1.3.4 Running in production

Recommended settings

Most default setting values in the application code base are suitable for production.

But the set of settings mentionned below might deserve some review or adjustments:

```
cliquet.http_scheme = https
cliquet.paginate_by = 100
cliquet.batch_max_requests = 25
cliquet.delete_collection_enabled = false
cliquet.basic_auth_enabled = false
cliquet.storage_pool_maxconn = 50
cliquet.cache_pool_maxconn = 50
fxa-oauth.cache_ttl_seconds = 3600
```

note For an exhaustive list of available settings and their default values, refer to cliquet source code.

Monitoring

```
# Heka
cliquet.logging_renderer = cliquet.logs.MozillaHekaRenderer

# StatsD
cliquet.statsd_url = udp://carbon.server:8125
```

Application output should go to `stdout`, and message format should have no prefix string:

```
[handler_console]
class = StreamHandler
args = (sys.stdout,)
level = INFO
formater = heka

[formatter_heka]
format = %(message)s
```

Adapt the logging configuration in order to plug Sentry:

```
[loggers]
keys = root, sentry

[handlers]
keys = console, sentry

[formatters]
keys = generic

[logger_root]
level = INFO
handlers = console, sentry

[logger_sentry]
level = WARN
handlers = console
qualname = sentry.errors
propagate = 0

[handler_console]
class = StreamHandler
args = (sys.stdout,)
level = INFO
formater = heka

[formatter_heka]
format = %(message)s

[handler_sentry]
class = raven.handlers.logging.SentryHandler
args = ('http://public:secret@example.com/1',)
level = WARNING
formatter = generic

[formatter_generic]
format = %(asctime)s,%(msecs)03d %(levelname)-5.5s [%(name)s] %(message)s
datefmt = %H:%M:%S
```

PostgreSQL setup

In production, it is wise to run the application with a dedicated database and user.

```
postgres=# CREATE USER produser;
postgres=# CREATE DATABASE proddb OWNER produser;
CREATE DATABASE
```

The tables needs to be created with the *cliquet* tool.

```
$ cliquet --ini config/readinglist.ini migrate
```

note Alternatively the SQL initialization files can be found in the *cliquet* source code (`cliquet/cache/postgresql/schemal.sql` and `cliquet/storage/postgresql/schemal.sql`).

Running with uWsgi

To run the application using uWsgi, an **app.wsgi** file is provided. This command can be used to run it:

```
uwsgi --ini config/readinglist.ini
```

uWsgi configuration can be tweaked in the ini file in the dedicated **[uwsgi]** section.

Here's an example:

```
[uwsgi]
wsgi-file = app.wsgi
enable-threads = true
http-socket = 127.0.0.1:8000
processes =  3
master = true
module = readinglist
harakiri = 30
uid = readinglist
gid = readinglist
virtualenv = .
lazy = true
lazy-apps = true
```

To use a different ini file, the `READINGLIST_INI` environment variable should be present with a path to it.

Running with gevent

It is possible to use gevent, by adding this in the configuration:

```
readinglist.gevent_enabled = true
```

Gevent and psycogreen should be installed in the virtualenv for it to work properly:

```
.venv/bin/pip install gevent psycogreen
```

note Gevent support is known to have issues with Python 3, and as such, it is discouraged to use it in this environment.

1.4 Contributing

Thank you for considering to contribute to *Reading List*!

note No contribution is too small; please submit as many fixes for typos and grammar bloopers as you can!

note Open a pull-request even if your contribution is not ready yet! It can be discussed and improved collaboratively!

1.4.1 Run tests

```
make tests
```

1.4.2 Run load tests

From the `loadtests` folder:

```
make test SERVER_URL=http://localhost:8000
```

Run a particular type of action instead of random:

```
LOAD_ACTION=batch_create make test SERVER_URL=http://localhost:8000
```

(*See loadtests source code for an exhaustive list of available actions and their respective randomness.*)

1.4.3 IRC channel

Join `#storage` on `irc.mozilla.org`!

1.5 Changelog

This document describes changes between each past release.

1.5.1 2.1.0 (unreleased)

- Nothing changed yet.

1.5.2 2.0.0 (2015-07-22)

Upgraded to Cliquet 2.3+.

Breaking changes

- Now requires PostgreSQL 9.4+
- Endpoints now expect articles to be posted in a `data` attribute
- Endpoints responses now contain a `data` attribute
- Endpoints switched from `If-Modified-Since` and `If-Unmodified-Since` to Etags

1.5.3 1.7.0 (2015-04-15)

New feature

- Article `title` now allowed to be empty or null (#250 and #253) (Original https://bugzilla.mozilla.org/show_bug.cgi?id=1152915)

Bug fixes

- Fix `stored_on` not being forced on creation (fixes #215)

Internal changes

- Enabled Coveralls with failing tests if coverage less than 100%

1.5.4 1.6.0 (2015-04-10)

Deployment instructions

Some changes were introduced in database schema. Run schema migration command before starting the application:

```
cliquet --ini production.ini migrate
```

New features

- Add more info in heartbeat (fixes #229)
- Clarify conflict docs (#244)
- Clarify data model docs (#247)
- Add PostgreSQL schema migration system (mozilla-services/cliquet#139)

See every features brought by Cliquet 1.7

Bug fixes

- Fix login prompt when Basic Auth is disabled (#237)
- Fix random IndexError in load tests (#238)
- Fix smoke tests configuration reading
- Fix Heka logging format of objects (#199)
- Fix performance of record insertion using ordered index (mozilla-services/cliquet#138)
- Fix 405 errors not JSON formatted (mozilla-services/cliquet#88)

See every bug fixes brought by Cliquet 1.7

1.5.5 1.5.0 (2015-03-30)

New features

- Split schema initialization from application startup, using a command-line tool.

```
cliquet --ini production.ini init
```

Bug fixes

- Fix documentation about WSGI and Sentry
- Fix connection pool no being shared between cache and storage (mozilla-services/cliquet#176)

- Default connection pool size to 10 (instead of 50) (mozilla-services/cliquet#176)
- Warn if PostgreSQL session has not UTC timezone (mozilla-services/cliquet#177)

Internal changes

- Deprecated `cliquet.storage_pool_maxconn` and `cliquet.cache_pool_maxconn` settings (renamed to `cliquet.storage_pool_size` and `cliquet.cache_pool_size`)

1.5.6 1.4.0 (2015-03-27)

New features

- Smoke test of FxA authentication using Loads (#220)
- Mesure calls on the authentication policy (mozilla-services/cliquet#167)
- Force default pagination to 100 if not set in settings (#214)
- Add documentation about setting up Sentry loggers (#227)

Breaking changes

- Prefix statsd metrics with the value of *cliquet.statsd_prefix* or *cliquet.project_name* (mozilla-services/cliquet#162)
- *http_scheme* setting has been replaced by *cliquet.http_scheme* and *cliquet.http_host* was introduced (mozilla-services/cliquet#151, mozilla-services/cliquet#166)
- URL in the hello view now has version prefix (mozilla-services/cliquet#165)

Bug fixes

- Fix changing read position (#213)
- Fix some PostgreSQL connection bottlenecks (mozilla-services/cliquet#170)
- Pull monitoring dependencies during install (#218)

Internal changes

- Update of PyFxA to get it working with gevent monkey patching (mozilla-services/cliquet#168)

1.5.7 1.3.0 (2015-03-25)

Deployment instructions

Until the database schema migration system is released (mozilla-services/cliquet#139), changes on schema have to be applied manually:

```
ALTER FUNCTION as_epoch(TIMESTAMP) IMMUTABLE;
CREATE INDEX idx_records_last_modified_epoch ON records(as_epoch(last_modified));
CREATE INDEX idx_deleted_last_modified_epoch ON deleted(as_epoch(last_modified));
```

New features

- Add setting to enable to asynchronous PostgreSQL using Psycogreen. (*default: disabled*). See installation documentation for more details on this.
- Add ability to execute only action in loads tests using the `LOAD_ACTION` environment variable. See contributing documentation for more details (#208).
- Add new load tests with several kinds of batch operations (#204)

Bug fixes

- Fix pagination URL in Next-page headers (fixes #210)
- Fix regression on records URL unicity when using ujson (#205)
- Fix hashing of user_id for BasicAuth (mozilla-services/cliquet#128)
- Force PostgreSQl session timezone to UTC (mozilla-services/cliquet#122)
- Make sure the *paginate_by* setting overrides the passed *limit* argument (mozilla-services/cliquet#129)
- Fix limit comparison under Python3 (mozilla-services/cliquet#143)
- Do not serialize using JSON if not necessary (mozilla-services/cliquet#131)
- Fix crash of classic logger with unicode (mozilla-services/cliquet#142)
- Fix crash of CloudStorage backend when remote returns 500 (mozilla-services/cliquet#142)
- Fix behaviour of CloudStorage with backslashes in querystring (mozilla-services/cliquet#142)
- Fix python3.4 segmentation fault (mozilla-services/cliquet#142)
- Add missing port in Next-Page header (mozilla-services/cliquet#147)

Internal changes

- Use postgres cache in loads tests (#203)
- Use ujson again, it was removed in the 1.3.2 release (#132)
- Add index for as_epoch(last_modified) (#130). Please add the following statements to SQL for the migration:
- Prevent fetching to many records for one user collection (#130)
- Use UPSERT for the heartbeat (#141)
- Improve tests of basic auth (#128)

1.5.8 1.2.0 (2015-03-20)

New features

- Add PostgreSQL connection pooling, with new settings `cliquet.storage_pool_maxconn` and `cliquet.cache_pool_maxconn` (*Default: 50*) (mozilla-services/cliquet#112)
- Add StatsD support, enabled with `cliquet.statsd_url = udp://server:port` (mozilla-services/cliquet#114)
- Add Sentry support, enabled with `cliquet.sentry_url = http://user:pass@server/1` (mozilla-services/cliquet#110)

Bug fixes

- Fix FxA verification cache not being used (mozilla-services/cliquet#103)
- Fix heartbeat database check (mozilla-services/cliquet#109)
- Fix PATCH endpoint crash if request has no body (mozilla-services/cliquet#115)

Internal changes

- Switch to ujson for JSON de/serialization optimizations (mozilla-services/cliquet#108)
- Use async connections for psycopg (#201)
- Imrpove the documentation layout (#200)

1.5.9 1.1.0 (2015-03-18)

Breaking changes

- *cliquet.storage.postgresql* now uses UUID as record primary key (mozilla-services/cliquet#70)
- Settings `cliquet.session_backend` and `cliquet.session_url` were renamed `cliquet.cache_backend` and `cliquet.cache_url` respectively.
- FxA user ids are not hashed anymore (mozilla-services/cliquet#82)
- Setting `cliquet.retry_after` was renamed `cliquet.retry_after_seconds`
- OAuth2 redirect url now requires to be listed in `fxa-oauth.webapp.authorized_domains` (e.g. `*.mozilla.com`)
- Batch are now limited to 25 requests by default (mozilla-services/cliquet#90)
- OAuth relier has been disabled by default (#193)

New features

- Every setting can be specified via an environment variable (e.g. `cliquet.storage_url` with `CLIQUET_STORAGE_URL`)
- Logging now relies on structlog (mozilla-services/cliquet#78)
- Logging output can be configured to stream JSON (mozilla-services/cliquet#78)
- New cache backend for PostgreSQL (mozilla-services/cliquet#44)
- Documentation was improved on various aspects (mozilla-services/cliquet#64, mozilla-services/cliquet#86)
- Handle every backend errors and return 503 errors (mozilla-services/cliquet#21)
- State verification for OAuth2 dance now expires after 1 hour (mozilla-services/cliquet#83)
- Add the preview field for an article (#156)
- Setup the readinglist OAuth scope (#16)
- Add a uwsgi file (#180)

Bug fixes

- FxA OAuth views errors are now JSON formatted (mozilla-services/cliquet#67)
- Prevent error when pagination token has bad format (mozilla-services/cliquet#72)
- List of CORS exposed headers were fixed in POST on collection (mozilla-services/cliquet#54)
- Fix environment variables not overriding configuration (mozilla-services/cliquet#100)
- Got rid of custom *CAST* in PostgreSQL storage backend to prevent installation errors without superuser (ref #174, mozilla-services/cliquet#99)

1.5.10 1.0 (2015-03-03)

Breaking changes

- Most configuration entries were renamed, see *config/readinglist.ini* example to port your configuration
- Status field was removed, archived and deleted fields were added (requires a database flush.)
- Remove Python 2.6 support

New features

- Add the /fxa-oauth/params endpoint
- Add the DELETE /articles endpoint (Needs cliquet.delete_collection_enabled configuration)
- Add the Response-Behavior header on PATCH /articles
- Add HTTP requests / responses examples in the documentation
- Use Postgresql as the default database backend

Internal changes

- Main code base was split into a separate project Cliquet
- Perform continuated pagination in loadtests
- Use PostgreSQL for loadtests

1.5.11 0.2.2 (2015-02-13)

Bug fixes

- Fix CORS preflight request permissions (PR #119)

1.5.12 0.2.1 (2015-02-11)

Breaking changes

- Internal user ids for FxA are now prefixed, all existing records will be lost (refs #109)

Bug fixes

- Fix CORS headers on validation error responses (ref #104)
- Fix handling of defaults in batch requests (ref #111, #112)

1.5.13 0.2 (2015-02-09)

Breaking changes

- PUT endpoint was disabled (ref #42)
- `_id` field was renamed to `id` (ref PR #91)
- FxA now requires a redirection URL (ref PR #69)

New features

- URLs uniques by user (ref #20)
- Handle conflicts responses (ref #45)
- Conditional changes for some articles attributes (ref #6)
- Batching support (ref #2)
- Pagination support (ref #25)
- Online documentation available at https://readinglist.readthedocs.io (ref PR #73)
- Basic Auth nows support any user/password combination (ref PR #78)

Bug fixes

- `marked_read_by` was ignored on PATCH (ref PR #72)
- Timestamp was not incremented on DELETE (ref PR #95)
- Fix number of bugs regarding support of CORS in error views (ref PR #105)
- Previous Basic Auth could impersonate FxA user (ref PR #78)

1.5.14 0.1 (2015-01-30)

- Allow Cors (#67)
- Log incomming request to the console (#65)
- Add timestamp for 304 and 412 response (#40)
- Add time vector to GET /articles and GET /articles/<id> (#4)
- Preconditions Headers for Update and Creation (#60)
- Provide number of items in headers of GET /articles (#39)
- Check for filter values (#58)
- Handle article title length (#37)
- Support min, max and no keywords filters (#43)
- Prevent to modify read-only fields (#26)
- Filtering and sort querystring (#44)
- Redis storage (#50)
- Handle errors (#24 - #49)
- Add loadtests (#47)
- Handle API version in URL (#33)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search