Workgroup: JMAP
Internet-Draft: draft-gondwana-jmap-blob-01
Updates: 8620 (if approved)
Published: 16 November 2020
Intended Status: Standards Track
Expires: 20 May 2021
Authors: B. Gondwana, Ed.
Fastmail

JMAP Blob management extension

Abstract

The JMAP base protocol (RFC8620) provides the ability to upload and download arbitrary binary data via HTTP PUT and GET on defined endpoint. This binary data is called a "Blob".

This extension adds additional ways to handle Blobs, by making inline method calls within a standard JMAP request.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 20 May 2021.

Table of Contents

1. Introduction

Sometimes JMAP ([RFC8620]) interactions require creating a Blob and then referencing it. In the same way that IMAP Literals ([RFC7888]) were extended to reduce roundtrips for simple data, embedding simple small blobs into the JMAP method stream can reduce roundtrips.

Likewise, when fetching an object, it can be useful to also fetch the raw content of that object without a separate roundtrip.

Where JMAP is being proxied through a system which is providing additional access restrictions, it can be useful to be able to see where a blob is referenced in order to decide whether to allow it to be downloaded.

2. Conventions Used In This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Blobs

A blob is a sequence of zero or more octets.

The JMAP base spec [RFC8210] defines the Blob/copy method, which is unchanged by this specfication.

3.1. Blob/set

This is a standard JMAP set method.

3.1.1. create

Properties:

Any one of:

*data:asText: String|null

*data:asBase64: String|null

*data:asHex: String|null

*catenate: [SetObject] *list of byte sources in order*

Also:

*type: String|null

Result is:

*id: Id the blobId

*type: String|null *as given in the creation (if any); or detected from content; or null*

*size: UnsignedInt *as per RFC8620 - the size of the file in Octets*

Any other properties identical to those that would be returned in the JSON response of the RFC8620 upload endpoint.

SetObject:

Any one of

*data:asText: String|null

*data:asBase64: String|null

*data:asHex: String|null

OR a blobId source:

*blobId: Id

*offset: UnsignedInt|null

*length: UnsignedInt|null

3.1.2. update

It is not possible to update a Blob, so any update will result in a notUpdated response.

3.1.3. destroy

If an uploaded Blob is not referenced by any persistent object, the server SHOULD destroy the object. Some systems use a content-based ID for blobs, so the server MAY respond destroyed and yet that blobId still exist with the same content.

Example:

Method Call:

```
[ "Blob/set", {
  "accountId" : "account1",
  "create" : {
    "1": {
      "data:asBase64": "iVBORw0KGgoAAAANSUhEUgAAAAEAAAABAQMAAAAl21bKA
                        AAAA1BMVEX/AAAZ4gk3AAAAAXRSTlN/gFy0ywAAAApJRE
                        FUeJxjYgAAAAYAAzY3fKgAAAAASUVORK5CYII=",
      "type" : "image/png"
    },
  },
}, "R1" ]
```

Response:

```
[ "Blob/set", {
  "accountId" : "account1",
  "created" : {
    "1": {
      "id" : "G4c6751edf9dd6903ff54b792e432fba781271beb",
      "type" : "image/png",
      "size" : 95
    },
  },
}, "R1" ]
```

3.2. Blob/get

A standard JMAP get.

Properties:

Any of

*data:asText

*data:asBase64

*data:asHex

*data *selects data:asText if the content is UTF-8, or data:asBase64*

*size

If not given, returns data and size.

QUESTION: do we want to add range operators?

*offset: UnsignedInt|null

*length: UnsignedInt|null

Returns that range of bytes (not characters!) from the blob

3.3. Blob/lookup

A reverse lookup!

Work to be done here, but something like this.

Map from blobId to object type:

e.g.

```
[ "Blob/lookup", {
  "objects": ["Mailbox", "Thread", "Email"],
  "ids": ["Gd2f81008cf07d2425418f7f02a3ca63a8bc82003",
          "G6f954bcb620f7f50fc8f21426bde3669da3d9067"]
}, "R1" ]
```

Response:

```
[ "Blob/lookup", {
  "list": [
    {
      "id": "Gd2f81008cf07d2425418f7f02a3ca63a8bc82003",
      "Mailbox": ["M54e97373", Mcbe6b662"],
      "Thread": ["T1530616e"],
      "Email": ["E16e70a73eb4", "E84b0930cf16"]
    },
  ],
  "notFound": ["G6f954bcb620f7f50fc8f21426bde3669da3d9067"]
}, "R1"]
```

This tells which objects of each type "contain" a reference to that blobId. "Contain" is defined somewhat losely here, so for example "the Mailbox contains an Email which references this blobId" is the standard in the response above, likewise for Thread.

4. Security considerations

TO BE IMPROVED:

JSON parsers are not all consistent in handling non-UTF-8 data. JMAP requires that all JSON data be UTF-8 encoded, so servers MUST either return data:asBase64 or isEncodingProblem: true and modify the data to be UTF-8 safe.

5. IANA considerations

TBD

6. Acknowledgements

TBD

7. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8210] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1", RFC 8210, DOI 10.17487/RFC8210, September 2017, <https://www.rfc-editor.org/info/rfc8210>.

[RFC8620] Jenkins, N. and C. Newman, "The JSON Meta Application Protocol (JMAP)", RFC 8620, DOI 10.17487/RFC8620, July 2019, <https://www.rfc-editor.org/info/rfc8620>.

8. Informative References

[RFC7888] Melnikov, A., Ed., "IMAP4 Non-synchronizing Literals", RFC 7888, DOI 10.17487/RFC7888, May 2016, <https://www.rfc-editor.org/info/rfc7888>.

Author's Address

Bron Gondwana (editor)
Fastmail
Level 2, 114 William St
Melbourne VIC 3000
Australia

Email: brong@fastmailteam.com
URI: https://www.fastmail.com