HTTP
Internet-Draft
Intended status: Experimental
Expires: May 24, 2020

A. Wright
November 21, 2019

Reporting Progress of Long-Running Operations in HTTP
draft-wright-http-progress-02

Abstract

This document defines a mechanism for following the real-time progress of long-running operations over HTTP.

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 May 24, 2020.

Table of Contents

1. Introduction

HTTP is often used for making and observing the progress of long-running operations, including:

- Copying, patching, or deleting large sets of files
- Waiting on a task to be started at a specific time
- Adding an operation to a lengthy queue
- Working through a multi-step operation, e.g. provisioning a server
- Receiving updates to a long running task, e.g. construction of a building

This document specifies a way to receive updates from the server on progress of such an operation, by defining a "progress" HTTP preference indicating the client would prefer to receive regular progress updates, a header for describing the current progress, and a 1xx interim response to convey this progress information.

1.1. Notational Conventions

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.

This document uses ABNF as defined in [RFC5234] and imports grammar rules from [RFC7230] and [RFC8187].

Examples in this document may add whitespace for clarity, or omit some HTTP headers for brevity; requests and responses may require additional Host, Connection, and/or Content-Length headers to be properly received.

1.2. Scope

This document is only intended to provide a mechanism for relaying the progress of a long-running operation, it does not intend to be a mechanism for subscribing to updates on a resource in general.

2. Status Document Workflow

The Status Document Workflow uses a status document that is related to a single request. This status document is updated with the status of the operation, until the operation completes, finalizing the status document with the result of the operation. No format is defined for the status document, any suitable information may be included, and the contents MAY be content-negotiated.

The server SHOULD keep the status document available for a period of time after the operation finishes.

2.1. Initial Request

To begin, the client makes the initial request with an unsafe method. For example, "POST http://example.com/resource".

- If the operation finishes quickly, the server can issue the final response with a non-1xx, non-202 status code. The server may respond with any response allowed by HTTP, including a document describing the result of the operation, a representation of the new state of the resource, or a minimal representation.

- If the client sent a "Prefer: processing" preference, the server SHOULD issue a "102 Processing" interim response upon receipt of the request, and every time there is an update to the operation

progress. The first interim response SHOULD include a "Location" header identifying the status document created for this request. When the request finishes, respond normally with the final non-1xx, non-202 status code.

- If the request includes "Prefer: respond-async, wait=n", and has been running longer than the preferred wait time, then background the operation and emit "202 Accepted", with a "Location" header. If the server emitted a 102 Processing interim response, this will be the same header as before.

If the server responds with the result of the operation, or a representation of the new state of the resource, the "Content-Location" header identifies where this document can be requested in the future.

Note that clients may make requests with all of the above preferences; they can all be honored at the same time, see below for an example.

2.2. Status Document Request

If the client received an operation status document from the initial unsafe request, it may make a GET request to this document to re-download the result of the request.

The client may do this for any reason, including:

- The operation resulted in a 202 Accepted response and the client wants to know if the operation finished.
- The user wants to review the outcome of the request after having discarded the initial 2xx (non-202) response.
- The connection was reset before the initial request could respond with a non-1xx status code.

If the client makes this request with the "Prefer: processing" preference, the server SHOULD send an initial "102 Processing" header, and "102 Processing" responses for every progress update until the operation completes.

2.3. Closing the Operation

The client MAY acknowledge it has reacted to the completed operation by issuing a "DELETE" request on the status document. Servers SHOULD limit requests on the status document to the user that issued the initial request.

Servers MAY delete the status document any time after the operation finishes, but SHOULD wait a period of time long enough for clients to check back on the operation on another business day.

2.4. Example

Clients may send any combination of preferences in a request. In this example, the client issues a POST request to capture a photograph of a scenic landscape by issuing a POST request to "http://example.com/capture", and the server generates a status document for this request at "http://example.com/capture?request=42".

```
POST http://example.com/capture HTTP/1.1
Prefer: processing, respond-async, wait=20
```

To which the server might reply:

```
HTTP/1.1 102 Processing
Location: <?request=42>
Progress: 0/3 "Herding cats"

HTTP/1.1 102 Processing
Progress: 1/3 "Knitting sweaters"

HTTP/1.1 102 Processing
Progress: 2/3 "Slaying dragons"

HTTP/1.1 201 Created
Progress: 3/3 "Available"
Location: </photos/42>
Content-Location: <?request=42>
Content-Type: text/plain

The photographer uploaded your image to:
  <http://example.com/photos/42>
```

If this same request took significantly longer (more than 20 seconds), then due to the respond-async preference, the response might look like this instead:

```
HTTP/1.1 102 Processing
Progress: 0/3 "Herding cats"
Location: </status>

HTTP/1.1 102 Processing
Progress: 1/3 "Knitting sweaters"

HTTP/1.1 202 Accepted
Location: </status>
Content-Location: </status>
Content-Type: text/plain

The photographer is on step 2: Knitting sweaters
```

The client can re-subscribe to updates by making a GET request to the status document with "Prefer: processing":

```
GET http://example.com/capture?request=42 HTTP/1.1
Prefer: processing, respond-async, wait=20

HTTP/1.1 102 Processing
Progress: 1/3 "Knitting sweaters"

HTTP/1.1 102 Processing
Progress: 2/3 "Slaying dragons"

HTTP/1.1 200 OK
Progress: 3/3 "Available"
Status-URI: 201 </capture>
Content-Type: text/plain

The photographer uploaded your image to:
  <http://example.com/photos/42>
```

3. Definitions

3.1. The "102 Processing" status code

The 102 (Processing) status code is an interim response used to inform the client that the server has accepted the request, but has not yet completed it. This status code SHOULD send this status when the request could potentially take long enough to time out connections due to inactivity, or when there is new progress to report via a "Progress" or "Status-URI" header.

The "102 Processing" status was first described by WebDAV in [RFC2518], but was not included in subsequent revisions of WebDAV for

lack of implementations. This document updates the semantics of the "102 Processing" status code first defined there.

3.1.1. Use of the "Location" header in 102 Processing

The meaning of a Location header [RFC7231] is the same as in a "202 Accepted" response: It identifies a document that will be updated with the progress, current status, and result of the operation.

A Location header SHOULD be sent in the first "102 Processing" response, as well as the "202 Accepted" response to the same request.

3.2. The "Progress" header

The "Progress" header is used to indicate the current progress on an operation being run by the origin server. Use of this header implies the server supports "102 Processing" responses and the "processing" preference.

```
Progress        = fraction *( WS progress-remark )
progress-remark = fraction / comment / quoted-string / ext-value
fraction        = 1*DIGIT "/" [ 1*DIGIT ]
comment         = <comment, see [RFC7230], Section 3.2.6>
quoted-string   = <quoted-string, see [RFC7230], Section 3.2.6>
ext-value       = <ext-value, see [RFC8187]>
```

The Progress header lists data about the current operation and summarizes operations that have finished. It contains a fraction, and any number of remarks.

The fraction numerator specifies the number of operations that have completed. It may also represent the zero-indexed identifier of the current operation. The numerator MUST NOT decrease in value.

The fraction denominator specifies the total expected operations to be completed before a final status code can be delivered. If specified, the denominator MUST NOT be smaller than the numerator. The denominator MAY be omitted when the length of the operation is unknown. If additional tasks need to be performed, the denominator MAY increase. The numerator MUST NOT decrease in value and MUST NOT disappear once introduced.

The remark is some sort of indication of the current task being carried out. For example, if multiple files are being operated on, it might refer to the most recent file to be opened. Four forms are provided:

- Use of additional "fraction" productions are permitted to indicate progress on a subordinate operation. For example, a data transfer in progress as part of a multi-step operation.

- Use of the "comment" production implies the text is not intended for end users.

- The "ext-value" provides a label for users. If the HTTP server supports localization, the server SHOULD negotiate a language using "Accept-Language", if it exists in the request. This language does not necessarily have to be the same as the "Content-Language".

- The "quoted-string" is also supported if the text is entirely 7-bit ASCII. This is suitable for reporting filenames or similar data not in any particular language.

Multiple remarks MAY be used. Remarks MUST be listed in descending significance; if multiple fractions are presented, remarks describe the operation identified by the previous fraction.

Example usage:

```
Progress: 0/1
Progress: 66/ (tries) utf-8'en'Generating%20prime%20number
Progress: 5/16 UTF-8'ja-JP'%e9%a3%9f%e3%81%b9%e3%81%a6
Progress: 3/20 "POST http://example.com/item/3" 8020/8591489 (bytes)
```

3.3. The "Status-URI" header

The Status-URI header reports the status of an operation performed on a resource by another request.

The Status-URI header MAY be used any number of times in a "101 Processing" response to report the result of a subordinate operation for the request.

```
Status-URI    = #status-pair
status-pair   = status-code OWS "<" URI-Reference ">"
status-code   = <status-code, see [RFC7230], Section 3.1.2>
URI-Reference = <URI-reference, see [RFC7230], Section 2.7>
```

Example usage:

```
Status-URI: 507 <http://example.com/photo/41>
Status-URI: 200 <http://example.com/capture>
```

3.4. The "processing" preference

The "processing" HTTP preference [RFC7240] specifies if the server should emit "102 Processing" status responses.

When performing a unsafe action, the server should emit interim "102 Processing" responses until the action finishes.

In a GET or HEAD request to a status document, it means the client is only interested in the result of the operation that the status document is about, and the server should send "102 Processing" updates until then. The "respond-async" and "wait" preferences are ignored here as the request is not performing an action.

4. Security Considerations

4.1. Status URIs

The fact that this operation produces a URI for each operation means that third parties can look at the requests being made by a user. Servers SHOULD ensure that only the user who made the request has access to the status document. Servers SHOULD generate URIs with sufficient entropy, although URIs supposed to be considered public knowledge (see HTTP).

4.2. Denial of Service

This may expose information about load, which may allow attackers to better exploit weak points already under stress. Servers with this functionality may make it cheap for server operators to accept work-intensive tasks. Usual precautions about mitigating denial-of-service attacks should be exercised.

5. References

5.1. 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>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7240] Snell, J., "Prefer Header for HTTP", RFC 7240, DOI 10.17487/RFC7240, June 2014, <https://www.rfc-editor.org/info/rfc7240>.

[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>.

[RFC8187] Reschke, J., "Indicating Character Encoding and Language for HTTP Header Field Parameters", RFC 8187, DOI 10.17487/RFC8187, September 2017, <https://www.rfc-editor.org/info/rfc8187>.

5.2. Informative References

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., Carter, S., and D. Jensen, "HTTP Extensions for Distributed Authoring -- WEBDAV", RFC 2518, DOI 10.17487/RFC2518, February 1999, <https://www.rfc-editor.org/info/rfc2518>.

Author's Address

Austin Wright

Email: aaa@bzfx.net