Network Working Group
Internet-Draft
Intended status: Experimental
Expires: January 1, 2015

E. Prud'hommeaux
W3C
June 30, 2014

The Hypertext Transfer Protocol (HTTP) Status Code 2NN (Contents of Related)
draft-prudhommeaux-http-status-2nn-00

Abstract

This document specifies the additional HyperText Transfer Protocol (HTTP) Status Code 2NN (Contents of Related). It also specified a Prefer header value "contents-of-related" which clients can use to indicate that they can accept 2NN responses.

Editorial Note (To be removed by RFC Editor before publication)

Distribution of this document is unlimited. Comments should be sent to the W3C Technical Architecture Group mailing list at www-tag@w3.org [1] (public archive [2]) and the Linked Data Platform mailing list at public-ldp-comments@w3.org [3] (public archive [4]). The latter list may be joined by sending a message with subject "subscribe" to public-ldp-comments-request@w3.org [5].

XML versions, latest edits, and the issues list for this document are available from [6].

Test cases related to redirection in general and the status code 2NN in particular can be found at [7] as a template.

Status of This Memo

This Internet-Draft will expire on January 1, 2015.

Copyright Notice

Table of Contents

1. Introduction

HTTP 2xx status codes indicate that the client's request was successfully received, understood, and accepted. The 2NN status code response asserts that Location field identifies a resource related to the requested resource and that the response contents are a

representation of that related resource. The 2NN response bypasses the extra round trip required for use cases conventionally solved with a 303 (See Other) response followed by the client performing a second GET on the target of that redirect. For example, 2NN streamlines these interactions which conventionally involve a server response with a Location header referencing the information needed by the client:

- An HTTP client performs a GET on a resource which is not an information resource. The server responds with a 303 and the client performs a second GET to retrieve an information resource related to the previous resource. (This idiom is frequently used to provide information about a resource while keeping that resource distinct from any page describing it.)

- An HTTP server responds to a POST request by creating a new resource and returning a 303 to redirect the client to that new resource. (This use case is described in [8] .)

- The resource requested in a GET is prohibitively large to serve and the server instead responds with a redirect to the beginning of a series of resources paginating the initial resource. The paginating resources are interlinked with the 'prev' and 'next' link headers described in [9] .

- A client has requested a Web application and the server responds with a multi-document response including e.g. HTML, images, CSS, Javascript and data for the web application.

- A client performs a POST which creates a new resource. The server has requested a Web application and the server responds with a multi-document response including e.g. HTML, images, CSS, Javascript and data for the web application.

2. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. 2NN Contents of Related

The 2NN (Contents of Related) status code indicates that the server is providing a response for the request method (e.g. GET or POST) performed on the URI in the Location header field, henceforth called the "related resource". The "expected response" is the response that the client would have received had it performed a GET on the related

resource. If the initial request method is HEAD, the expected response has no message body (see RFC 7231 4.3.2. HEAD [10]).

By returning a 2NN status code, the server asserts that the expected response has a status code of 200, and that its response (with the 2NN) is the same as the expected response with the status code changed to 2NN and a Location header added to identify the related resource. As with Content-Location, such a claim can only be trusted if both identifiers share the same resource owner, which cannot be programmatically determined via HTTP (see RFC 7231 3.1.4.2. Content-Location [11]).

For caching purposes, see Section 3.1 below. For purposes other than caching, the response is interpreted as if the response code were 200 and the effective request URI were the related resource. This defines the semantics for all current headers other than Location, as well as future headers defined as extensions to HTTP 1.1. A 2NN MUST NOT be used if the expected response includes a Location header.

The following example demonstrates the use of 2NN responses to streamline the creation of new resources as described by [LDP]. The 2NN response is generic; it can be used for any use case where the server expects a client to dereference a Location header, for example, image tiling or packaging web applications.

Client request:

```
  GET /bigDoc HTTP/1.1
  Host: bigco.example
  Accept: text/turtle, q=1.0; application/rdf+xml, q=0.9
  Prefer: contents-of-related
```

Server 2NN response:

```
 HTTP/1.1 2NN Contents of Related
 Content-Type: text/turtle
 Location: http://bigco-static.example/p1
 Link: <http://bigco-static.example/p2>; rel="next"
 Content-Location: http://bigco-static.example/p1.ttl
 Content-Length: 145

 <http://bigco.example/bigDoc> <http://purl.org/dc/terms/description>
           "Here is everything we know about this giant resource...".
```

Here, the related resource is http://bigco-static.example/p1 and the expected response is same as the Server 2NN response above, but with a 200 status code and no Location header. The above example

communicates the same response as the following client-server exchanges where the client performs on operation on a resource, the server responds with a 303, and the client performs a GET (or HEAD) on the resource in the Location header of the servers 303 response:

Client request:

```
  GET /bigDoc HTTP/1.1
  Host: bigco.example
  Accept: text/turtle, q=1.0; application/rdf+xml, q=0.9
```

Server 303 response:

```
 HTTP/1.1 303 See Related
 Content-Type: text/html
 Location: http://bigco-static.example/p1
 Content-Length: 125

 <html><head><title>303</title></head><body><p>
 You probably want <a href="http://bigco-static.example/p1">this</a>.
 </p></body></html>
```

Client request on the "related resource":

```
  GET /p1 HTTP/1.1
  Host: bigco.example
  Accept: text/turtle, q=1.0; application/rdf+xml, q=0.9
```

Server response (defined as the "expected response"):

```
 HTTP/1.1 200 OK
 Content-Type: text/turtle
 Link: <http://bigco-static.example/p2>; rel="next"
 Content-Location: http://bigco-static.example/p1.ttl
 Content-Length: 145

 <http://bigco.example/bigDoc> <http://purl.org/dc/terms/description>
           "Here is everything we know about this giant resource...".
```

Note that in the Server 2NN response above, the Content-Location provides a content-negotiated representation of the requested resource and the Link provides paging information. Both illustrate how a 2NN response header (other than Location) is interpreted as applying to the resource in the Location header, http://bigco-static.example/p1 in this example.

3.1. Caching Semantics

The client and any intervening proxies SHOULD cache the 2NN response for the original effective request URI. If the client has out of band reason to trust the server's claim that a GET performed on the value of the Location header would have elicited the same response, they may additionally cache a 200 response for a GET on value of the Location header.

In the example Server 2NN response above, the client and intervening proxies should cache the 2NN response to the GET of http://bigco.example/bigDoc with the associated Accept header. If the client has out of band knowledge that bigco.example has some authority to answer for http://bigco-static.example/p1 and http://bigco-static.example/p1.ttl , it may associate the expected response with those resources as well.

4. contents-of-related Prefer header value

Per [12], this document registers the Prefer header ([RFC7240]) value "contents-of-related". A client MAY include a "Prefer: contents-of-related" header with a request to indicate that the client can accept 2NN responses.

5. Deployment Considerations

Section 4 of [13] specified that all 2xx status codes indicate a successful request. However, some conventional clients may not be specifically programmed to accept content accompanying a 2xx response other than 200. Therefore, initial use of status code 2NN will be restricted to cases where the server has sufficient confidence in the clients understanding the new code. The contents-of-related Prefer header value (see Section 4) is one way for the client to advertise its support for 2NN responses.

6. Security Considerations

All security considerations that apply to either 303 or 200 response codes apply also to the 2NN status code (see Section 12 of [RFC7231]). Additionally, indiscriminately caching the 2NN response as the response to the related resource permits malicious or irresponsible servers to poison cache entries for 3rd parties. See RFC 7231 [14] for similar constraints about associating cache entries with the value of a Content-Location header. In particular, the caching semantics including the warning "can only be trusted if both identifiers share the same resource owner, which cannot be programmatically determined via HTTP."

7. IANA Considerations

The registration below shall be added to the HTTP Status Code Registry (defined in Section 4.2 of [RFC7231] and located at [15]):

Value	Description	Reference
2NN	Contents of Related	Section 3 of this specification

8. Acknowledgements

The definition for the new status code 2NN re-uses text from the HTTP/1.1 definitions of 2xx status codes. The structure and much of the text of this draft was taken from [16]. John Arwe, Jenni Tennison, and the W3C TAG and Linked Data Working Group for excellent input and review.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC7231] Fielding, R., Ed., Lafon, Y., Ed., and J. Reschke, Ed., "HTTP/1.1, part 2: Message Semantics", RFC 7231, March 2012.

[RFC7240] Snell, J., Ed., "HTTP/1.1, part 2: Message Semantics", RFC 7240, June 2012.

9.2. Informative References

[LDP] Speicher, S., Ed., Arwe, J., Ed., and A. Malhotra, Ed., "Linked Data Platform 1.0", W3C Candidate Recommendation CR-ldp-20140619, June 2014, <http://www.w3.org/turtle>.

Latest version available at [17].

9.3. URIs

[1] http://tools.ietf.org/html/rfc7231#section-6.4.4

[2] https://tools.ietf.org/html/rfc5005#section-3

[3] http://tools.ietf.org/html/rfc7231#section-4.3.2

[4] http://tools.ietf.org/html/rfc7231#section-3.1.4.2

[5] http://tools.ietf.org/html/rfc5226#section-5

[6] http://tools.ietf.org/html/rfc7231#section-3.1.4.2

[7] http://tools.ietf.org/html/rfc7231#section-3.1.4.2

[8] http://www.iana.org/assignments/http-status-codes

[9] http://tools.ietf.org/html/draft-reschke-http-status-308-07

Appendix A. Implementations (to be removed by RFC Editor before publication)

@@Expected from W3C Linked Data Platform Working Group

Appendix B. Change Log (to be removed by RFC Editor before publication)

B.1. No previous version

...

Appendix C. Resolved issues (to be removed by RFC Editor before publication)

Issues that were either rejected or resolved in this version of this document.

C.1. noPreviousVersion

no previous versions

...

Appendix D. Open issues (to be removed by RFC Editor prior to publication)

D.1. edit

Type: edit

eric@w3.org (2014-02-21): Umbrella issue for editorial fixes/enhancements.

Author's Address

Eric G. Prud'hommeaux
World Wide Web Consortium
32 Vassar St.
Cambridge, MA 02140
USA

EMail: eric@w3.org
URI: http://www.w3.org/People/Eric/ericP-foaf#ericP