Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 30 December 2025

M. Nottingham
28 June 2025

HTTP Availability Hints
draft-nottingham-http-availability-hints-02

Abstract

This specification defines availability hints, a new class of HTTP responses headers that augment the information in the Vary header field.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-nottingham-http-availability-hints/.

information can be found at https://mnot.github.io/I-D/.

Source for this draft and an issue tracker can be found at https://github.com/mnot/I-D/labels/availability-hints.

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 30 December 2025.

Table of Contents

1. Introduction

The HTTP Vary header field (Section 12.5.5 of [HTTP]) allows an origin server to describe what aspects of requests affect the content of its responses. This information is useful in many ways: most prominently, downstream caches can use it to select the correct stored response for a given request (Section 4.1 of [HTTP-CACHING]).

However, the information conveyed by Vary is limited. If the request headers enumerated in it are considered as a n-dimensional space with each field representing an axis, this response header:

Vary: Accept-Encoding, Accept-Language, ECT

indicates that there is a three-dimensional space of potential responses that could be sent, but nothing more is conveyed. The number and nature of the entries on each axis are not available, leaving caches and other downstream consumers none the wiser as to how broad this space is, or how to navigate it.

This design makes using Vary difficult. A cache doesn't have enough information available to decide whether one of its stored responses is the best to satisfy a given request in all but the most simple circumstances.

For example, if a request indicates that the client prefers responses in the French language, but will also accept English, and the cache has a stored English response, what is the appropriate action? Should it serve the English response, or should it make a request to the server for a French response and hope that one might be available -- adding significant latency if it is not?

This specification defines a new type of HTTP header field -- an _availability hint_ -- that augments the information on a single axis of content negotiation, by describing the selection of responses that a server has available along that axis. So, our example above could have three availability hints added to it:

```
Vary: Accept-Encoding, Accept-Language, ECT
Avail-Encoding: gzip, br
Avail-Language: fr, en;d
Avail-ECT: ("slow-2g" "2g" "3g"), ("4g");d
```

This says that there are two encodings available -- gzip and brotli -- beyond the mandatory "identity" encoding; that both French and English are available, but English is the default; and that there are two different representations available depending on the Effective Connection Type that the client advertises, with "4g" being the default. (note that because it it not yet standardised, this specification does not define the Avail-ECT header field; it is only used as an example above).

Caches and other clients can use this information to determine when a request can be satisfied by a stored response, and what other options might be available. Using the example above, we can know that the response to a request an ECT of "2g" can also be used for a request with "3g".

Availability hints have some limitations. While a server's preferences along a single axis of negotiation can be conveyed by the corresponding availability hint, its relative preferences between multiple axes are not. In the example above, it isn't possible to

know whether the server prefers that downstream caches and clients use the brotli-encoded French version over the gzip-encoded English version.

Likewise, it is’t possible to convey "holes" in the dimensional space described by Vary. For example, a gzip-encoded French response may not be available from the server. This specification does not attempt to address this shortcoming.

Availability hints do not specify exactly how caches should behave in all circumstances. Because they operate as an optimisation, they often have different behaviours based upon the specific requirements of their deployment. Availability hints are designed to better inform their operation, not constrain it.

Finally, availability hints need to be defined for each axis of content negotiation in use, and the recipient (such as a cache) needs to understand that type of availability hint. If either condition is not true, that axis of negotiation will fall back to the behaviour specified by Vary.

Section 2 describes how availability hints are defined. Section 3 specifies how availability hints are processed, with respect to the Vary header field. Section 4 defines a number of availability hints for existing HTTP content negotiation mechanisms.

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.

2. Defining Availability Hints

An availability hint applies to a single axis of HTTP proactive content negotiation; for example, that enabled by the Accept-Encoding request header field.

A specification for an availability hint needs to convey the following information:

1. The definition of a response header field that describes the available responses along that axis of content negotiation. By convention, these header fields typically begin with "Avail-".

2. An algorithm or guidelines for using that information to determine whether a stored response can be selected for a presented request (per Section 4.1 of [HTTP-CACHING]).

The response header field should be defined as a Structured Field [STRUCTURED-FIELDS].

It is recommended that the selection algorithm operate solely using information in the stored responses and presented request, if possible. If the selection algorithm can return multiple available responses, they should indicate an order of preference.

Either the response header field or the algorithm should indicate which of the available responses is the default -- i.e., which is used if none match.

3. Calculating Cache Keys with Availability Hints

When a cache is presented with a response that has both a Vary header field and one or more availability hints, this specification augments the process defined in Section 4.1 of [HTTP-CACHING].

While Vary processing is defined in terms of whether the header fields from two requests match, availability hints invoke a different processing model. When they are present, the set of stored responses that can be used to satisfy a presented request is found by following these steps:

1. Determine the Vary header field and availability hints present for the requested URL. They SHOULD be obtained from the most recently obtained response for that URL (per Section 4 of [HTTP-CACHING], but without applying the Vary header field per Section 4.1 of [HTTP-CACHING]), although they MAY be obtained from any response allowed to be used for that URL.

2. For each content negotiation axis in the Vary header field, determine which stored responses can be selected by running the corresponding selection algorithm, as defined by the associated availability hint.

    1. If an axis of content negotiation is not recognised or implemented by the cache, fall back to selecting available responses for that axis using the rules described in Section 4.1 of [HTTP-CACHING].

3. Return the intersection of the results of (2).

This specification does not define how to select the most appropriate response when more than one is returned, but it is RECOMMENDED that client preferences be observed when expressed.

If a received availability hint does not conform to the data structure defined in its specification (for example, a Structured Fields List of Strings is anticipated, but one item in the List is an Integer), that field SHOULD be ignored (with the implication that processing for that axis of content negotiation will fall back to Vary behavior). Note that unrecognised Parameters on Structured Fields are ignored and therefore do not trigger this fallback, unless explicitly specified otherwise.

4. Availability Hint Definitions

The following subsections define availability hints for a selection of existing content negotiation mechanisms.

4.1. Content Encoding

The Avail-Encoding response header field is the availability hint for content negotiation using the Accept-Encoding request header field defined in Section 12.5.3 of [HTTP]. For example:

```
Vary: Accept-Encoding
Avail-Encoding: gzip, br
```

Avail-Encoding is a Structured Field, whose value is a List (Section 3.1 of [STRUCTURED-FIELDS]) of Tokens (Section 3.3.4 of [STRUCTURED-FIELDS]). Each list item indicates a content-coding that is available for the resource. Additionally, the "identity" encoding is always considered to be available, and is the default encoding.

The selection algorithm for this axis of content negotiation is described in Section 12.5.3 of [HTTP].

4.2. Content Format

The Avail-Format response header field is the availability hint for content negotiation using the Accept request header field defined in Section 12.5.1 of [HTTP]. For example:

```
Vary: Accept
Avail-Format: image/png, image/gif;d
```

Avail-Format is a Structured Field, whose value is a List (Section 3.1 of [STRUCTURED-FIELDS]) of Tokens (Section 3.3.4 of [STRUCTURED-FIELDS]). Each list item indicates a media type ("type/subtype") that is available for the resource.

A single member of the List MAY have the "d" parameter, which indicates that member is the default format.

The selection algorithm for this axis of content negotiation is described in Section 12.5.1 of [HTTP].

4.3. Content Language

The Avail-Language response header field is the availability hint for content negotiation using the Accept-Language request header field defined in Section 12.5.4 of [HTTP]. For example:

```
Vary: Accept-Language
Avail-Language: en-uk, en-us;d, fr, de
```

Avail-Format is a Structured Field, whose value is a List (Section 3.1 of [STRUCTURED-FIELDS]) of Tokens (Section 3.3.4 of [STRUCTURED-FIELDS]). Each list item indicates a language tag that is available for the resource.

A single member of the List MAY have the "d" parameter, which indicates that member is the default language.

The selection algorithm for this axis of content negotiation is described in Section 12.5.4 of [HTTP].

4.4. Cookie

The Cookie-Indices response header field is the availability hint for content negotiation using the Cookie request header field defined in [RFC6265]. For example:

```
Vary: Cookie
Cookie-Indices: "id", "sid"
```

Cookie-Indices is a Structured Field, whose value is a List (Section 3.1 of [STRUCTURED-FIELDS]) of Strings (Section 3.3.3 of [STRUCTURED-FIELDS]). Each list item indicates a cookie name whose value is to be considered when selecting responses. Unlisted cookies are, by implication, ignored for that purpose.

The selection algorithm for Cookie-Indices, given a set of stored_responses a presented_request, and the value of Cookie-Indices:

1. Let available_responses be an empty set.

2. For each stored_response in stored_responses:

    1. For each cookie_index in Cookie-Indices:

        1. Let presented_values be a list of the values of the cookies with the name cookie_index in presented_request, lexicographically sorted. If no cookie with the name cookie_index is present in presented_request, let presented_values be an empty list.

        2. Let stored_values be a list of the values of the cookies with the name cookie_index in stored_response, lexicographically sorted. If no cookie with the name cookie_index is present in stored_response, let stored_values be an empty list.

        3. If presented_values does not equal stored_values, continue to the next stored_response.

    2. Add stored_response to available_responses.

3. Return available_responses.

Note that this algorithm requires storing the cookies from the associated request with each response.

5. IANA Considerations

IANA is asked to update the "Hypertext Transfer Protocol (HTTP) Field Name Registry" registry according as follows:

Field Name	Status	Structured Type	Reference
Avail-Encoding	permanent	List	Section 4.1
Avail-Format	permanent	List	Section 4.2
Avail-Language	permanent	List	Section 4.3
Cookie-Indices	permanent	List	Section 4.4

Table 1

6. Security Considerations

Availability hints reveal information about the state of the server, and therefore could help an attacker understand better how to bypass defenses. For example, Cookie-Indices reveals information about what cookies might affect the content of responses.

Recipients of availability hints should not assume that the information within them are complete or correct.

7. References

7.1. Normative References

[HTTP] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[HTTP-CACHING] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Caching", STD 98, RFC 9111, DOI 10.17487/RFC9111, June 2022, <https://www.rfc-editor.org/rfc/rfc9111>.

[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/rfc/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/rfc/rfc8174>.

[STRUCTURED-FIELDS]
Nottingham, M. and P. Kamp, "Structured Field Values for HTTP", RFC 9651, DOI 10.17487/RFC9651, September 2024, <https://www.rfc-editor.org/rfc/rfc9651>.

7.2. Informative References

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, DOI 10.17487/RFC6265, April 2011, <https://www.rfc-editor.org/rfc/rfc6265>.

Author’s Address

Mark Nottingham
Melbourne
Australia
Email: mnot@mnot.net
URI: https://www.mnot.net/