HTTP Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 27, 2016

I. Grigorik
Google

November 24, 2015

HTTP Client Hints
draft-ietf-httpbis-client-hints-00

Abstract

An increasing diversity of Web-connected devices and software capabilities has created a need to deliver optimized content for each device.

This specification defines a set of HTTP request header fields, colloquially known as Client Hints, to address this. They are intended to be used as input to proactive content negotiation; just as the Accept header allows clients to indicate what formats they prefer, Client Hints allow clients to indicate a list of device and agent specific preferences.

Note to Readers

The issues list for this draft can be found at https://github.com/httpwg/http-extensions/labels/client-hints .

Status of This Memo

Table of Contents

1. Introduction

There are thousands of different devices accessing the web, each with different device capabilities and preference information. These device capabilities include hardware and software characteristics, as well as dynamic user and client preferences.

One way to infer some of these capabilities is through User-Agent (UA) detection against an established database of client signatures. However, this technique requires acquiring such a database, integrating it into the serving path, and keeping it up to date.

However, even once this infrastructure is deployed, UA sniffing has numerous limitations:

- UA detection cannot reliably identify all static variables
- UA detection cannot infer any dynamic client preferences
- UA detection requires an external device database
- UA detection is not cache friendly

A popular alternative strategy is to use HTTP cookies to communicate some information about the client. However, this approach is also not cache friendly, bound by same origin policy, and imposes additional client-side latency by requiring JavaScript execution to create and manage HTTP cookies.

This document defines a set of new request header fields that allow the client to perform proactive content negotiation [RFC7231] by indicating a list of device and agent specific preferences, through a mechanism similar to the Accept header which is used to indicate preferred response formats.

Client Hints does not supersede or replace the User-Agent header field. Existing device detection mechanisms can continue to use both mechanisms if necessary. By advertising its capabilities within a request header field, Client Hints allows for cache friendly and proactive content negotiation.

1.1. 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].

This document uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234] with the list rule extension defined in [RFC7230], Appendix B. It includes by reference the DIGIT rule from [RFC5234]; the OWS, field-name and quoted-string rules from [RFC7230]; and the

parameter rule from [RFC7231].

2. Client Hint Request Header Fields

A Client Hint request header field is a HTTP header field that is used by HTTP clients to indicate configuration data that can be used by the server to select an appropriate response. Each one conveys a list of client preferences that the server can use to adapt and optimize the response.

This document defines a selection of Client Hint request header fields, and can be referenced by other specifications wishing to use the same syntax and processing model.

2.1. Sending Client Hints

Clients control which Client Hint headers and their respective header fields are communicated, based on their default settings, user configuration and/or preferences. The user may be given the choice to enable, disable, or override specific hints.

The client and server, or an intermediate proxy, may use an opt-in mechanism to negotiate which fields should be reported to allow for efficient content adaption.

2.2. Server Processing of Client Hints

Servers MAY respond with an optimized response based on one or more received hints from the client. When doing so, and if the resource is cacheable, the server MUST also emit a Vary response header field ([RFC7234]), and optionally Key ([I-D.ietf-httpbis-key]), to indicate which hints were used and whether the selected response is appropriate for a later request.

Further, depending on the used hint, the server MAY also need to emit additional response header fields to confirm the property of the response, such that the client can adjust its processing. For example, this specification defines "Content-DPR" response header field that MUST be returned by the server when the "DPR" hint is used

to select the response.

2.2.1. Advertising Support for Client Hints

Servers can advertise support for Client Hints using the Accept-CH header or an equivalent HTML meta element with http-equiv attribute.

```
Accept-CH = #token
```

For example:

```
Accept-CH: DPR, Width, Viewport-Width, Downlink
```

When a client receives Accept-CH, it SHOULD append the Client Hint headers that match the advertised field-values. For example, based on Accept-CH example above, the client would append DPR, Width, Viewport-Width, and Downlink headers to all subsequent requests.

2.2.2. Interaction with Caches

When selecting an optimized response based on one or more Client Hints, and if the resource is cacheable, the server MUST also emit a Vary response header field ([RFC7234]) to indicate which hints were used and whether the selected response is appropriate for a later request.

```
Vary: DPR
```

Above example indicates that the cache key should be based on the DPR header.

```
Vary: DPR, Width, Downlink
```

Above example indicates that the cache key should be based on the DPR, Width, and Downlink headers.

Client Hints MAY be combined with Key ([I-D.ietf-httpbis-key]) to enable fine-grained control of the cache key for improved cache

efficiency. For example, the server MAY return the following set of instructions:

```
Key: DPR;partition=1.5:2.5:4.0
```

Above example indicates that the cache key should be based on the value of the DPR header with three segments: less than 1.5, 1.5 to less than 2.5, and 4.0 or greater.

```
Key: Width;div=320
```

Above example indicates that the cache key should be based on the value of the Width header and be partitioned into groups of 320: 0-320, 320-640, and so on.

```
Key: Downlink;partition=0.5:1.0:3.0:5.0:10
```

Above example indicates that the cache key should be based on the (Mbps) value of the Downlink header with six segments: less than 0.5, 0.5 to less than 1.0, 1.0 to less than 3.0, 3.0 to less than 5.0, 5.0 to less than 10; 10 or higher.

3. The DPR Client Hint

The "DPR" header field is a number that, in requests, indicates the client's current Device Pixel Ratio (DPR), which is the ratio of physical pixels over CSS px of the layout viewport on the device.

```
DPR = 1*DIGIT [ "." 1*DIGIT ]
```

If DPR occurs in a message more than once, the last value overrides all previous occurrences.

3.1. Confirming Selected DPR

The "Content-DPR" header field is a number that indicates the ratio

between physical pixels over CSS px of the selected image response.

```
  Content-DPR = 1*DIGIT [ "." 1*DIGIT ]
```

DPR ratio affects the calculation of intrinsic size of image resources on the client - i.e. typically, the client automatically scales the natural size of the image by the DPR ratio to derive its display dimensions. As a result, the server must explicitly indicate the DPR of the selected image response whenever the DPR hint is used, and the client must use the DPR value returned by the server to perform its calculations. In case the server returned Content-DPR value contradicts previous client-side DPR indication, the server returned value must take precedence.

Note that DPR confirmation is only required for image responses, and the server does not need to confirm the resource width as this value can be derived from the resource itself once it is decoded by the client.

If Content-DPR occurs in a message more than once, the last value overrides all previous occurrences.

4. The Width Client Hint

The "Width" header field is a number that, in requests, indicates the resource width in physical px (i.e. intrinsic size of an image). The provided physical px value is a number rounded to the largest smallest following integer (i.e. ceiling value).

```
  Width = 1*DIGIT
```


If the resource width is not known at the time of the request or the resource does not have a display width, the Width header field may be omitted. If Width occurs in a message more than once, the last value overrides all previous occurrences.

5. The Viewport-Width Client Hint

The "Viewport-Width" header field is a number that, in requests,

indicates the layout viewport width in CSS px. The provided CSS px value is a number rounded to the largest smallest following integer (i.e. ceiling value).

```
Viewport-Width = 1*DIGIT
```

If Viewport-Width occurs in a message more than once, the last value overrides all previous occurrences.

6. The Downlink Client Hint

The "Downlink" header field is a number that, in requests, indicates the client's maximum downlink speed in megabits per second (Mbps), as defined by the "downlinkMax" attribute in the W3C Network Information API.

```
Downlink = 1*DIGIT [ "." 1*DIGIT ]
```

If Downlink occurs in a message more than once, the minimum value should be used to override other occurrences.

7. The Save-Data Hint

The "Save-Data" header field is a token that, in requests, indicates client's preference for reduced data usage, due to high transfer costs, slow connection speeds, or other reasons.

```
Save-Data = "on"
```

The token is a signal indicating explicit user opt-in into a reduced data usage mode on the client, and when communicated to origins allows them to deliver alternate content honoring such preference - e.g. smaller image and video resources, alternate markup, and so on.

8. Examples

For example, given the following request headers:

```
DPR: 2.0
Width: 320
Viewport-Width: 320
```

The server knows that the device pixel ratio is 2.0, that the intended display width of requested resource is 160 CSS px (320 physical pixels at 2x resolution), and that the viewport width is 320 CSS px.

If the server uses above hints to perform resource selection for an image asset, it must confirm its selection via the Content-DPR response header to allow the client to calculate the appropriate intrinsic size of the image response. The server does not need to confirm resource width, only the ratio between physical pixels and CSS px of the selected image resource:

```
Content-DPR: 1.0
```

The Content-DPR response header indicates to the client that the server has selected resource with DPR ratio of 1.0. The client may use this information to perform additional processing on the resource - for example, calculate the appropriate intrinsic size of the image resource such that it is displayed at the correct resolution.

Alternatively, the server could select an alternate resource based on the maximum downlink speed advertised in the request headers:

```
Downlink: 0.384
```

The server knows that the client's maximum downlink speed is 0.384Mbps (GPRS EDGE), and it may use this information to select an optimized resource - for example, an alternate image asset, stylesheet, HTML document, media stream, and so on.

9. Security Considerations

Client Hints defined in this specification do not expose any new information about the user's environment beyond what is already available to, and may be communicated by, the application at runtime via JavaScript - e.g. viewport and image display width, device pixel ratio, and so on.

However, implementors should consider the privacy implications of various methods to enable delivery of Client Hints - see "Sending Client Hints" section. For example, sending Client Hints on all requests may make information about the user's environment available to origins that otherwise did not have access to this data (e.g. origins hosting non-script resources), which may or not be the desired outcome. The implementors may want to provide mechanisms to control such behavior via explicit opt-in, or other mechanisms. Similarly, the implementors should consider how and whether delivery of Client Hints is affected when the user is in "incognito" or similar privacy mode.

10. IANA Considerations

This document defines the "Accept-CH", "DPR", "Width", and "Downlink" HTTP request fields, "Content-DPR" HTTP response field, and registers them in the Permanent Message Header Fields registry.

- Header field name: DPR
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints
- Header field name: Width
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints
- Header field name: Viewport-Width
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints
- Header field name: Downlink
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints
- Header field name: Content-DPR
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF

- Specification document(s): [this document]
- Related information: for Client Hints

- Header field name: Accept-CH
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints
- Header field name: Save-Data
- Applicable protocol: HTTP
- Status: standard
- Author/Change controller: IETF
- Specification document(s): [this document]
- Related information: for Client Hints

11. Normative References

[I-D.ietf-httpbis-key]
Fielding, R. and m. mnot, "The Key HTTP Response Header Field", draft-ietf-httpbis-key-00 (work in progress), October 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://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, <http://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, <http://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, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <http://www.rfc-editor.org/info/rfc7234>.

Author's Address

Ilya Grigorik
Google

Email: ilya@igvita.com
URI: https://www.igvita.com/