HTTP M.
West
Internet-Draft
Google
Intended status: Standards Track November 29,
2018
Expires: June 2, 2019

The 'Lang' Client Hint
draft-west-lang-client-hint-00

Abstract

This document defines a Client Hint that aims to allow developers to opt-in to the ability to perform content negotiation based on the set of natural languages preferred by the user agent. This new mechanism is intended to improve upon the privacy properties of the "Accept-Language" header, and eventually to supplant it entirely.

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 http://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 June 2, 2019.

Table of Contents

1. Introduction

Today, user agents generally specify a set of preferred languages as part of each HTTP request by sending the "Accept-Languages" header along with each request (defined in Section 5.3.5 of [RFC7231]). This header aims to give servers the opportunity to serve users the best content available in a language they understand. For example,

my browser currently sends the following header:

```
Accept-Language: en-US, en;q=0.9, de;q=0.8
```

This tells the server something along the lines of "This user prefers American English, but will accept any English at all. If no English-languge content is available, try German!". If the server has English-language content, it might redirect the user agent to that preferred content. If not, it might try German.

In the best case, this kind of content negotiation sincerely improves the user experience, giving them legible content they enjoy reading. This comes with a cost, however, as language preferences are fairly unique, and end up exposing quite a bit of entropy to the web.

This document proposes a mechanism that might allow user agents to reduce the passive fingerprinting surface exposed by the "Accept-

Language" header by replacing it with a new "Sec-CH-Lang" Client Hint ([I-D.ietf-httpbis-client-hints]) that servers can opt-into receiving. Rather than broadcasting this information to everyone on the network, all the time, user agents can make reasonable decisions about how to respond to given sites' requests for language preferences.

1.1. Example

A user navigates to "https://example.com/" for the first time. Their user agent sends no language preferences along with the HTTP request. The server, however, is interested in rendering content consistent with the users' preferences, and requests this data by sending an "Accept-CH" header (Section 2.2.1 of [I-D.ietf-httpbis-client-hints]) along with the response:

```
Accept-CH: Lang
```

In response, the user agent includes language preferences in subsequent requests:

```
Sec-CH-Lang: "en-US", "en", "de"
```

1.2. 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. The 'Lang' Client Hint

The 'Lang' Client Hint exposes a user agent's language preferences to a server. The definitions below assume that each user agent has defined a "preferred languages list", which contains an arbitrary number of strings adhering to the "language-range" grammar defined in Section 2.1 of [RFC4647], and which is sorted in descending order of user preference. The example given above, for instance, might result in the list << "en-US", "en", "de" >>.

2.1. The 'Sec-CH-Lang' Header Field

The "Sec-CH-Lang" request header field gives a server information

about a user agent's language preferences. It is a Structured Header
([I-D.ietf-httpbis-header-structure]) whose value MUST be a list
([I-D.ietf-httpbis-header-structure], Section 3.2). Each item in the

list MUST be a string ([I-D.ietf-httpbis-header-structure], Section 3.7).

The header's ABNF is:

```
Sec-CH-Arch = sh-list
```

To generate a "Sec-CH-Lang" header value for a given request, user agents MUST:

1. If the request's client-hints set includes "Lang", then:

    1. Let "value" be a Structured Header whose value is an empty list.

    2. For each item in the user agent's "preferred languages list":

        1. Append the item to "value".

    3. Set a header in request's header list whose name is "Sec-CH-Lang", and whose value is "value".

2.2. Integration with Fetch

The Fetch specification should call into the following algorithm in place of the current Step 1.4 in its HTTP-network-or-cache fetch algorithm.

To set the language metadata for a request ("r"), the user agent MUST execute the following steps:

1. If request's header list does not contain "Accept-Language", then the user agent MAY append a header whose name is "Accept-Language" and whose value corresponds to the requirements in Section 5.3.5 of [RFC7231] to "request"'s header list.

2. Set request's "Sec-CH-Lang" header, as described in Section 2.1.

3. Security and Privacy Considerations

3.1. Secure Transport

Client Hints will not be delivered to non-secure endpoints (see the secure transport requirements in Section 2.2.1 of [I-D.ietf-httpbis-client-hints]). This means that language preferences will not be leaked over plaintext channels, reducing the opportunity for network attackers to build a profile of a given agent's behavior over time.

3.2. Delegation

Client Hints will be delegated from top-level pages via Feature Policy (once a few patches against Fetch and Client Hints and Feature Policy land. This reduces the likelihood that language preferences will be delivered along with subresource requests, which reduces the potential for passive fingerprinting.

- Fetch integration of Accept-CH opt-in: https://github.com/whatwg/fetch/issues/773
- HTML integration of Accept-CH-Lifetime and the ACHL cache: https://github.com/whatwg/html/issues/3774
- Adding new CH features to the CH list in Fetch: https://github.com/whatwg/fetch/issues/725
- Other PRs for adding the Feature Policy 3rd party opt-in: https://github.com/whatwg/fetch/issues/811 and https://github.com/wicg/feature-policy/issues/220

3.3. Access Restrictions

Language preferences expose quite a bit of entropy to the web. User agents ought to exercise judgement before granting access to this information, and MAY impose restrictions above and beyond the secure transport and delegation requirements noted above. For instance, user agents could choose to deliver the "Sec-CH-Lang" header only on navigation, but not on subresource requests. Likewise, they could offer users control over the values revealed to servers, or gate access on explicit user interaction via a permission prompt or via a settings interface.

4. Implementation Considerations

4.1. The 'Accept-Language' Header

User agents SHOULD deprecate the "Accept-Language" header in favor of the Client Hints model described in this document. This deprecation can take place in stages, perhaps by first limiting the scopes in which the header is sent (navigations not subresources, etc), but the goal should be to remove the header entirely in favor of this opt-in model.

4.2. The 'Sec-CH-' prefix

Based on some discussion in https://github.com/w3ctag/design-reviews/issues/320, it seems reasonable to forbid access to these headers from JavaScript, and demarcate them as browser-controlled client hints so they can be documented and included in requests without triggering CORS preflights. A "Sec-CH-" prefix seems like a viable approach, but this bit might shift as the broader Client Hints discussions above coalesce into something more solid that lands in specs.

4.3. What about weight?

The "Accept-Language" header includes an optional weight along with each listed language (e.g. "en;q=0.3" is less preferred than "de;q=0.9"). A potential application of that is expressing equal preference for two or more languages, but the challenge of exposing such an option to users (compared to an ordered list) seems to make practical use unlikely. Moreover, widely-used implementations of the "Accept-Language" header blindly assign weights in exactly this way (see Chromium's "HttpUtil::GenerateAcceptLanguageHeader", and Firefox's "rust_prepare_accept_languages").

In this document, I'm boldly (foolishly?) asserting that "q" weighting can be removed without impact, in favor of assigning semantic meaning to the ordering of the items in the header list.

5. IANA Considerations

This document intends to define the "Sec-CH-Lang" HTTP request header field, and to register it in the permanent message header field registry ([RFC3864]).

It also intends to deprecate the "Accept-Language" header field.

5.1. 'Sec-CH-Lang' Header Field

Header field name: Sec-CH-Lang

Applicable protocol: http

Status: standard

Author/Change controller: IETF

Specification document: this specification (Section 2.1)

5.2. 'Accept-Language' Header Field

Header field name: Accept-Language

Applicable protocol: http

Status: deprecated

Author/Change controller: IETF

Specification document: this specification (Section 4.1), and Section 5.3.5 of [RFC7231]

6. Normative References

[I-D.ietf-httpbis-client-hints] Grigorik, I., "HTTP Client Hints", draft-ietf-httpbis-client-hints-06 (work in progress), July 2018.

[I-D.ietf-httpbis-header-structure] Nottingham, M. and P. Kamp, "Structured Headers for HTTP", draft-ietf-httpbis-header-structure-08 (work in progress), October 2018.

[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>.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, DOI 10.17487/RFC3864, September 2004, <https://www.rfc-editor.org/info/rfc3864>.

[RFC4647] Phillips, A. and M. Davis, "Matching of Language Tags", BCP 47, RFC 4647, DOI 10.17487/RFC4647, September 2006, <https://www.rfc-editor.org/info/rfc4647>.

[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>.

[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>.

Appendix A. Changes

A.1. draft-west-ua-client-hints-00

- This specification sprang, fully-formed, from the head of Zeus.

Author's Address

Mike West
Google

Email: mkwst@google.com