HTTPAPI
Internet-Draft
Intended status: Standards Track
Expires: 13 June 2024

S. Dalal

E. Wilde

11 December 2023

The Deprecation HTTP Header Field
draft-ietf-httpapi-deprecation-header-03

Abstract

The Deprecation HTTP response header field is used to signal to consumers of a URI-identified resource that the resource will be or has been deprecated. Additionally, the deprecation link relation can be used to link to a resource that provides additional information about planned or existing deprecation, and possibly ways in which clients can best manage deprecation.

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-ietf-httpapi-deprecation-header/.

Discussion of this document takes place on the HTTPAPI Working Group mailing list (mailto:httpapi@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/httpapi/. Subscribe at https://www.ietf.org/mailman/listinfo/httpapi/. Working Group information can be found at https://ietf-wg-httpapi.github.io/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-httpapi/deprecation-header.

Status of This Memo

Table of Contents

1. Introduction

Deprecation of an HTTP resource (Section 3.1 of [HTTP]) communicates information about the lifecycle of a resource. It encourages applications to migrate away from the resource, discourages applications from forming new dependencies on the resource, and informs applications about the risk of continued dependence upon the resource.

The act of deprecation does not change any behavior of the resource. It informs clients of the fact that a resource will be or is deprecated. The Deprecation HTTP response header field can be used to convey this at runtime to clients and carries information indicating when the deprecation will be in effect.

In addition to the Deprecation header field, the resource provider can use other header fields to convey additional information related to deprecation. This can be information such as where to find documentation related to the deprecation, what can be used as a replacement, and when a deprecated resource becomes non-operational.

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 specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234] and includes, by reference, the sf-date format as defined in [SFBIS].

The term "resource" is to be interpreted as defined in Section 3.1 of [HTTP].

2. The Deprecation HTTP Response Header Field

The Deprecation HTTP response header field allows a server to communicate to a client that the resource in context of the message is or will be deprecated.

2.1. Syntax

The Deprecation response header field describes the deprecation of the resource identified with the response it occurred within (see Section 6.4.2 of [HTTP]). It conveys the deprecation date, which may be in the future (the resource context will be deprecated at that date) or in the past (the resource context has been deprecated at that date). Deprecation is an Item Structured Header [RFC8941]. Refer to Section 3.3.7 of [SFBIS] for ABNF of sf-date:

```
Deprecation = sf-date
```

Servers MUST NOT include more than one Deprecation header field in the same response.

The date is the date when the resource was or will be deprecated. It is in the form of an Structured Field Date as defined in Section 3.3.7 of [SFBIS].

The following example shows that the resource context has been deprecated on Friday, June 30, 2023 at 23:59:59 GMT:

```
Deprecation: @1688169599
```

The deprecation date can be in the future. This means that the resource will be deprecated at the indicated date in the future.

2.2. Scope

The Deprecation header field applies to the resource identified with the response it occurred within (see Section 6.4.2 of [HTTP]), meaning that it announces the upcoming deprecation of that specific resource. However, there may be scenarios where the scope of the announced deprecation is larger than just the single resource where it appears.

Resources are free to define such an increased scope, and usually this scope will be documented by the resource so that consumers of the resource know about the increased scope and can behave accordingly. When doing so, it is important to take into account that such increased scoping is invisible for consumers who are unaware of the increased scoping rules. This means that these consumers will not be aware of the increased scope, and they will not interpret deprecation information different from its standard meaning (i.e., it applies to the resource only).

Using such an increased scope still may make sense, as deprecation information is only a hint anyway. It is optional information that cannot be depended on, and clients should always be implemented in ways that allow them to function without Deprecation information. Increased scope information may help clients to glean additional hints from related resources and, thus, might allow them to implement behavior that allows them to make educated guesses about resources becoming deprecated.

For example, an API might not use Deprecation header fields on all of its resources, but only on designated resources such as the API's home document. This means that deprecation information is available, but in order to get it, clients have to periodically inspect the home document. In this example, the extended context of the Deprecation header field would be all resources provided by the API, while the visibility of the information would only be on the home document.

3. The Deprecation Link Relation Type

In addition to the Deprecation HTTP header field, the server can use links with the "deprecation" link relation type to communicate to the client where to find more information about deprecation of the context. This can happen before the actual deprecation, to make a deprecation policy discoverable, or after deprecation, when there may be documentation about the deprecation, and possibly documentation of how to manage it.

This specification places no restrictions on the representation of the linked deprecation policy. In particular, the deprecation policy may be available as human-readable documentation or as machine-readable description.

3.1. Documentation

The purpose of the Deprecation header field is to provide a hint about deprecation to the resource consumer. Upon reception of the Deprecation header field, the client developer can look up the resource's documentation in order to find deprecation related information. The resource provider can provide a link to the resource documentation using a Link header field with relation type deprecation as shown below:

```
Link: <https://developer.example.com/deprecation>;
      rel="deprecation"; type="text/html"
```

In this example the linked content provides additional information about deprecation of the resource context. There is no Deprecation header field in the response, and thus the resource is not (yet) deprecated. However, the resource already exposes a link where information is available how deprecation is managed for the resource context. This may be documentation explaining the use of the Deprecation header field, and also explaining under which circumstances and with which policies (announcement before deprecation; continued operation after deprecation) deprecation might be happening.

The following example uses the same link header field, but also announces a deprecation date using a Deprecation header field:

```
Deprecation: @1688169599
Link: <https://developer.example.com/deprecation>;
      rel="deprecation"; type="text/html"
```

Given that the deprecation date is in the past, the linked information resource may have been updated to include information about the deprecation, allowing consumers to discover information about the deprecation and how to best manage it.

3.1.1. Security Considerations

The Deprecation header field SHOULD be treated as a hint, meaning that the resource is indicating (and not guaranteeing with certainty) that it will be or is deprecated. Applications consuming the resource SHOULD check the resource documentation to verify authenticity and accuracy. Resource documentation SHOULD provide additional information about the deprecation, potentially including recommendation(s) for replacement.

In cases where the Deprecation header field value is a date in the future, it can lead to information that otherwise might not be available. Therefore, applications consuming the resource SHOULD verify the resource documentation and if possible, consult the resource developer to discuss potential impact due to deprecation and plan for possible transition to recommended resource.

In cases where a Link header field is used to provide documentation, one should assume that the content of the Link header field may not be secure, private or integrity-guaranteed, and due caution should be exercised when using it. Applications consuming the resource SHOULD check the referred resource documentation to verify authenticity and accuracy.

4. Sunset

In addition to the deprecation related information, if the resource provider wants to convey to the client application that the deprecated resource is expected to become unresponsive at a specific point in time, the Sunset HTTP header field [RFC8594] can be used in addition to the Deprecation header field.

The timestamp given in the Sunset header field MUST NOT be earlier than the one given in the Deprecation header field.

The following example shows that the resource in context has been deprecated since Friday, June 30, 2023 at 23:59:59 GMT and its sunset date is Sunday, June 30, 2024 at 23:59:59 GMT. Please note that for historical reasons the Sunset HTTP header field uses a different data type for date.

```
Deprecation: @1688169599
Sunset: Sun, 30 Jun 2024 23:59:59 GMT
```

5. Resource Behavior

The act of deprecation does not change any behavior of the resource. Deprecated resources SHOULD keep functioning as before, allowing consumers to still use the resources in the same way as they did before the resources were declared deprecated.

6. IANA Considerations

6.1. The Deprecation HTTP Response Header Field

The Deprecation response header field should be added to the "Hypertext Transfer Protocol (HTTP) Field Name Registry" registry (Section 16.3.1 of [HTTP])

Header Field Name: Deprecation

Applicable Protocol: Hypertext Transfer Protocol (HTTP)

Status: Standard

Author: Sanjay Dalal <sanjay.dalal@cal.berkeley.edu>,
Erik Wilde <erik.wilde@dret.net>

Change controller: IETF

Specification document: this specification,
Section 2 "The Deprecation HTTP Response Header Field"

6.2. The Deprecation Link Relation Type

The deprecation link relation type should be added to the permanent registry of link relation types (Section 4.2 of [LINK]).

Relation Type: deprecation

Applicable Protocol: Hypertext Transfer Protocol (HTTP)

Status: Standard

Author: Sanjay Dalal <sanjay.dalal@cal.berkeley.edu>,
Erik Wilde <erik.wilde@dret.net>

Change controller: IETF

Specification document: this specification,
Section 3 "The Deprecation Link Relation Type"

7. Examples

The following example does not show complete HTTP interaction. It only shows those HTTP header fields in a response that are relevant for resource deprecation.

```
Deprecation: @1688169599
Link: <https://developer.example.com/deprecation>; rel="deprecation"
```

8. References

8.1. 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>.

[LINK] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/rfc/rfc8288>.

[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>.

[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/rfc/rfc5234>.

[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, <https://www.rfc-editor.org/rfc/rfc7234>.

[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>.

[RFC8941] Nottingham, M. and P. Kamp, "Structured Field Values for HTTP", RFC 8941, DOI 10.17487/RFC8941, February 2021, <https://www.rfc-editor.org/rfc/rfc8941>.

[SFBIS] Nottingham, M. and P.-H. Kamp, "Structured Field Values for HTTP", 6 November 2023, <https://datatracker.ietf.org/doc/draft-ietf-httpbis-sfbis/>.

8.2. Informative References

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/rfc/rfc7942>.

[RFC8594] Wilde, E., "The Sunset HTTP Header Field", RFC 8594, DOI 10.17487/RFC8594, May 2019, <https://www.rfc-editor.org/rfc/rfc8594>.

Appendix A. Implementation Status

Note to RFC Editor: Please remove this section before publication.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

A.1. Implementing the Deprecation Header Field

This is a list of implementations that implement the deprecation header field:

Organization: Apollo

* Description: Deprecation header field is returned when deprecated functionality (as declared in the GraphQL schema) is accessed

* Reference: https://www.npmjs.com/package/apollo-server-tools

Organization: Zalando

* Description: Deprecation header field is recommended as the preferred way to communicate API deprecation in Zalando API designs.

* Reference: https://opensource.zalando.com/restful-api-guidelines/#deprecation

Organization: Palantir Technologies

* Description: Deprecation header field is incorporated in code generated by conjure-java, a CLI to generate Java POJOs and interfaces from Conjure API definitions

* Reference: https://github.com/palantir/conjure-java

Organization: IETF Internet Draft, Registration Protocols Extensions

* Description: Deprecation link relation is returned in Registration Data Access Protocol (RDAP) notices to indicate deprecation of jCard in favor of JSContact.

* Reference: https://tools.ietf.org/html/draft-loffredo-regext-rdap-jcard-deprecation

Organization: E-Voyageurs Technologies

* Description: Deprecation header field is incorporated in Hesperides, a configuration management tool providing universal

text file templating and properties editing through a REST API or a webapp.

* Reference: https://github.com/voyages-sncf-technologies/hesperides/blob/master/documentation/lightweight-architecture-decision-records/deprecated_endpoints.md

Organization: Open-Xchange

* Description: Deprecation header field is used in Open-Xchange appsuite-middleware

* Reference: https://github.com/open-xchange/appsuite-middleware

Organization: MediaWiki

* Description: Core REST API of MediaWiki would use Deprecation header field for endpoints that have been deprecated because a new endpoint provides the same or better functionality.

* Reference: https://phabricator.wikimedia.org/T232485

A.2. Implementing the Concept

This is a list of implementations that implement the general concept, but do so using different mechanisms:

Organization: Zapier

* Description: Zapier uses two custom HTTP header fields named X-API-Deprecation-Date and X-API-Deprecation-Info

* Reference: https://zapier.com/engineering/api-geriatrics/

Organization: IBM

* Description: IBM uses a custom HTTP header field named Deprecated

* Reference: https://www.ibm.com/support/knowledgecenter/en/SS42VS_7.3.1/com.ibm.qradar.doc/c_rest_api_getting_started.html

Organization: Ultipro

* Description: Ultipro uses the HTTP Warning header field as described in Section 5.5 of [RFC7234] with code 299

* Reference: https://connect.ultipro.com/api-deprecation

Organization: Clearbit

* Description: Clearbit uses a custom HTTP header field named X-API-

Warn

* Reference: https://blog.clearbit.com/dealing-with-deprecation/

Organization: PayPal

* Description: PayPal uses a custom HTTP header field named PayPal-Deprecated

* Reference: https://github.com/paypal/api-standards/blob/master/api-style-guide.md#runtime

Appendix B. Changes from Draft-02

This revision has made the following changes:

* Date format is changed from IMF-fixdate rule as defined in Section 5.6.7 of [HTTP] to Structured Field for Date as defined in Section 3.3.7 of [SFBIS].

Appendix C. Acknowledgments

The authors would like to thank Nikhil Kolekar, Darrel Miller, Mark Nottingham, and Roberto Polli for their contributions.

The authors take all responsibility for errors and omissions.

Authors' Addresses

Sanjay Dalal
Email: sanjay.dalal@cal.berkeley.edu
URI: https://github.com/sdatspun2

Erik Wilde
Email: erik.wilde@dret.net
URI: http://dret.net/netdret