HTTPbis Working Group
Internet-Draft
Updates: 2616 (if approved)
Intended status: Standards Track
Expires: March 7, 2011

J. Reschke
greenbytes

September 3, 2010

Use of the Content-Disposition Header Field in the Hypertext Transfer Protocol (HTTP) draft-ietf-httpbis-content-disp-00

Abstract

HTTP/1.1 defines the Content-Disposition response header field, but points out that it is not part of the HTTP/1.1 Standard. This specification takes over the definition and registration of Content-Disposition, as used in HTTP, and clarifies internationalization aspects.

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

This specification is expected to replace the definition of Content-Disposition in the HTTP/1.1 specification, as currently revised by the IETF HTTPbis working group. See also <http://www3.tools.ietf.org/wg/httpbis/trac/ticket/123>.

Discussion of this draft should take place on the HTTPBIS working group mailing list (ietf-http-wg@w3.org). The current issues list is at <http://trac.tools.ietf.org/wg/httpbis/trac/query?component=content-disp> and related documents (including fancy diffs) can be found at <http://tools.ietf.org/wg/httpbis/>.

Status of This Memo

Table of Contents

1. Introduction

HTTP/1.1 defines the Content-Disposition response header field in Section 19.5.1 of [RFC2616], but points out that it is not part of the HTTP/1.1 Standard (Section 15.5):

> Content-Disposition is not part of the HTTP standard, but since it is widely implemented, we are documenting its use and risks for implementers.

This specification takes over the definition and registration of Content-Disposition, as used in HTTP. Based on interoperability testing with existing User Agents, it fully defines a profile of the features defined in the Multipurpose Internet Mail Extensions (MIME) variant ([RFC2183]) of the header field, and also clarifies internationalization aspects.

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].

This specification uses the augmented BNF notation defined in Section 2.1 of [RFC2616], including its rules for linear whitespace (LWS).

3. Header Field Definition

The Content-Disposition response header field is used to convey

additional information about how to process the response payload, and also can be used to attach additional metadata, such as the filename.

3.1. Grammar

```
content-disposition = "Content-Disposition" ":"
                       disposition-type *( ";" disposition-parm )

disposition-type    = "inline" | "attachment" | disp-ext-type
                    ; case-insensitive
disp-ext-type       = token

disposition-parm    = filename-parm | disp-ext-parm

filename-parm       = "filename" "=" value
                    | "filename*" "=" ext-value

disp-ext-parm       = token "=" value
                    | ext-token "=" ext-value
ext-token           = <the characters in token, followed by "*">
```

Defined in [RFC2616]:

```
token       = <token, defined in [RFC2616], Section 2.2>
value       = <value, defined in [RFC2616], Section 3.6>
```

Defined in [RFC5987]:

```
ext-value   = <ext-value, defined in [RFC5987], Section 3.2>
```

3.2. Disposition Type

If the disposition type matches "attachment" (case-insensitively), this indicates that the user agent should not display the response, but directly enter a "save as..." dialog.

On the other hand, if it matches "inline" (case-insensitively), this implies default processing.

Other disposition types SHOULD be handled the same way as "attachment" (see also [RFC2183], Section 2.8).

3.3. Disposition Parameter: 'Filename'

The parameters "filename" and "filename*", to be matched case-insensitively, provide information on how to construct a filename for storing the message payload.

Depending on the disposition type, this information might be used right away (in the "save as..." interaction caused for the "attachment" disposition type), or later on (for instance, when the

user decides to save the contents of the current page being displayed).

"filename" and "filename*" behave the same, except that "filename*" uses the encoding defined in [RFC5987], allowing the use of characters not present in the ISO-8859-1 character set ([ISO-8859-1]). When both "filename" and "filename*" are present, a recipient SHOULD pick "filename*" and ignore "filename" - this will make it possible to send the same header value to clients that do not support "filename*".

It is essential that user agents treat the specified filename as advisory only, thus be very careful in extracting the desired information. In particular:

- When the value contains path separator characters, all but the last segment SHOULD be ignored. This prevents unintentional overwriting of well-known file system location (such as "/etc/passwd").

- Many platforms do not use Internet Media Types ([RFC2046]) to hold type information in the file system, but rely on filename extensions instead. Trusting the server-provided file extension could introduce a privilege escalation when later on the file is opened locally (consider ".exe"). Thus, recipients need to ensure that a file extension is used that is safe, optimally matching the media type of the received payload.

- Other aspects recipients need to be aware of are names that have a special meaning in the filesystem or in shell commands, such as "." and "..", "~", "|", and also device names.

3.4. Disposition Parameter: Extensions

To enable future extensions, unknown parameters SHOULD be ignored (see also [RFC2183], Section 2.8).

3.5. Extensibility

Note that Section 9 of [RFC2183] defines IANA registries both for disposition types and disposition parameters. This registry is shared by different protocols using Content-Disposition, such as MIME and HTTP. Therefore, not all registered values may make sense in the context of HTTP.

4. Examples

Direct UA to show "save as" dialog, with a filename of "foo.html":

Content-Disposition: Attachment; filename=foo.html

Direct UA to behave as if the Content-Disposition header field wasn't present, but to remember the filename "foo.html" for a subsequent

save operation:

Content-Disposition: INLINE; FILENAME= "foo.html"

Direct UA to show "save as" dialog, with a filename of "an example":

Content-Disposition: Attachment; Filename*=UTF-8'en'an%20example

Note that this example uses the extended encoding defined in [RFC5987] to specify that the natural language of the filename is English, and also to encode the space character which is not allowed in the token production.

Direct UA to show "save as" dialog, with a filename containing the Unicode character U+20AC (EURO SIGN):

Content-Disposition: attachment; filename*= UTF-8''%e2%82%ac%20rates

Here, the encoding defined in [RFC5987] is also used to encode the non-ISO-8859-1 character.

Same as above, but adding the "filename" parameter for compatibility with user agents not implementing RFC 5987:

```
Content-Disposition: attachment; filename="EURO rates";
                                 filename*=utf-8''%e2%82%ac%20rates
```

Note: as of August 2010, many user agents unfortunately did not properly handle unexpected parameters, and some that implement RFC 5987 did not pick the extended parameter when both were present.

5. Internationalization Considerations

The "filename*" parameter (Section 3.3), using the encoding defined in [RFC5987], allows the server to transmit characters outside the ISO-8859-1 character set, and also to optionally specify the language in use.

Future parameters might also require internationalization, in which case the same encoding can be used.

6. Security Considerations

Using server-supplied information for constructing local filenames introduces many risks. These are summarized in Section 3.3.

Furthermore, implementers also ought to be aware of the Security Considerations applying to HTTP (see Section 15 of [RFC2616]), and also the parameter encoding defined in [RFC5987] (see Appendix).

7. IANA Considerations

7.1. Registry for Disposition Values and Parameter

This specification does not introduce any changes to the registration procedures for disposition values and parameters that are defined in Section 9 of [RFC2183].

7.2. Header Field Registration

This document updates the definition of the Content-Disposition HTTP header field in the permanent HTTP header field registry (see [RFC3864]).

Header field name: Content-Disposition

Applicable protocol: http

Status: standard

Author/Change controller: IETF

Specification document: this specification (Section 3)

8. Acknowledgements

Thanks to Rolf Eike Beer, Alfred Hoenes, and Roar Lauritzsen for their valuable feedback.

9. References

9.1. Normative References

[ISO-8859-1] International Organization for Standardization, "Information technology -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1", ISO/IEC 8859-1:1998, 1998.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC5987] Reschke, J., "Applicability of RFC 2231 Encoding to Hypertext Transfer Protocol (HTTP) Headers", RFC 5987, August 2010.

9.2. Informative References

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[RFC2183] Troost, R., Dorner, S., and K. Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997.

[RFC2231] Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 3629, STD 63, November 2003.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", RFC 3986, STD 66, January 2005.

Appendix A. Changes from the RFC 2616 Definition

Compared to Section 19.5.1 of [RFC2616], the following normative changes reflecting actual implementations have been made:

- According to RFC 2616, the disposition type "attachment" only

applies to content of type "application/octet-stream". This restriction has been removed, because user agents in practice do

not check the content type, and it also discourages properly declaring the media type.

- RFC 2616 only allows "quoted-string" for the filename parameter. This would be an exceptional parameter syntax, and also doesn't reflect actual use.

- The definition for the disposition type "inline" ([RFC2183], Section 2.1) has been re-added with a suggestion for its processing.

- This specification requires support for the extended parameter encoding defined in [RFC5987].

Appendix B. Differences compared to RFC 2183

Section 2 of [RFC2183] defines several additional disposition parameters: "creation-date", "modification-date", "quoted-date-time", and "size". These do not appear to be implemented by any user agent, thus have been omitted from this specification.

Appendix C. Alternative Approaches to Internationalization

By default, HTTP header field parameters cannot carry characters outside the ISO-8859-1 ([ISO-8859-1]) character encoding (see [RFC2616], Section 2.2). For the "filename" parameter, this of course is an unacceptable restriction.

Unfortunately, user agent implementers have not managed to come up with an interoperable approach, although the IETF Standards Track specifies exactly one solution ([RFC2231], clarified and profiled for HTTP in [RFC5987]).

For completeness, the sections below describe the various approaches that have been tried, and explains how they are inferior to the RFC 5987 encoding used in this specification.

C.1. RFC 2047 Encoding

RFC 2047 defines an encoding mechanism for header fields, but this encoding is not supposed to be used for header field parameters - see Section 5 of [RFC2047]:

> An 'encoded-word' MUST NOT appear within a 'quoted-string'.
>
> ...

> An 'encoded-word' MUST NOT be used in parameter of a MIME Content-Type or Content-Disposition field, or in any structured field body except within a 'comment' or 'phrase'.

In practice, some user agents implement the encoding, some do not (exposing the encoded string to the user), and some get confused by it.

C.2. Percent Encoding

Some user agents accept percent encoded ([RFC3986], Section 2.1) sequences of characters encoded using the UTF-8 ([RFC3629]) character encoding.

In practice, this is hard to use because those user agents that do not support it will display the escaped character sequence to the user.

Furthermore, the first user agent to implement this did choose the encoding based on local settings; thus making it very hard to use in multi-lingual environments.

C.3. Encoding Sniffing

Some user agents inspect the value (which defaults to ISO-8859-1) and switch to UTF-8 when it seems to be more likely to be the correct interpretation.

As with the approaches above, this is not interoperable and furthermore risks misinterpreting the actual value.

C.4. Implementations

Unfortunately, as of August 2010, neither the encoding defined in RFCs 2231 and 5789, nor any of the alternate approaches discussed above was implemented interoperably. Thus, this specification recommends the approach defined in RFC 5987, which at least has the advantage of actually being specified properly.

The table below shows the implementation support for the various approaches: [[impls: Discuss: should we mention the implementation status of actual UAs in a RFC? Up to the IESG to decide...]]

User Agent	RFC 2231/5987	RFC 2047	Percent Encoding	Encoding Sniffing
Chrome	no	yes	yes	yes
Firefox	yes (*)	yes	no	yes
Internet Explorer	no	no	yes	no
Konqueror	yes	no	no	no
Opera	yes (*)	no	no	no
Safari	no	no	no	yes

(*) Does not implement the fallback behavior to "filename" described in Section 3.3.

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

D.1. Since draft-reschke-rfc2183-in-http-00

Adjust terminology ("header" -> "header field"). Update rfc2231-in-http reference.

D.2. Since draft-reschke-rfc2183-in-http-01

Update rfc2231-in-http reference. Actually define the "filename" parameter. Add internationalization considerations. Add examples using the RFC 5987 encoding. Add overview over other approaches, plus a table reporting implementation status. Add and resolve issue "nodep2183". Add issues "asciivsiso", "deplboth", "quoted", and "registry".

D.3. Since draft-reschke-rfc2183-in-http-02

Add and close issue "docfallback". Close issues "asciivsiso", "deplboth", "quoted", and "registry".

D.4. Since draft-reschke-rfc2183-in-http-03

Updated to be a Working Draft of the IETF HTTPbis Working Group.

Index

Author's Address

Julian F. Reschke
greenbytes GmbH
Hafenweg 16
Muenster, NW 48155
Germany

EMail: julian.reschke@greenbytes.de
URI: http://greenbytes.de/tech/webdav/