Use of the Content-Disposition Header Field in the
Hypertext Transfer Protocol (HTTP)
draft-ietf-httpbis-content-disp-05

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://trac.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/.
The changes in this draft are summarized in Appendix Appendix D.9.

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 August 21, 2011.

Copyright Notice

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.

> *Note: this document does not apply to Content-Disposition header
> fields appearing in message payloads transmitted over HTTP, such
> as when using the media type "multipart/form-data" ([RFC2388]).

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 implied linear whitespace
(LWS).

3. Conformance and Error Handling

This specification defines conformance criteria for both senders
(usually, HTTP origin servers) and recipients (usually, HTTP user
agents) of the Content-Location header field. An implementation is
considered conformant if it complies with all of the requirements
associated with its role.
This specification also defines certain forms of the header field-value
to be invalid, using both ABNF and prose requirements, but it does not
define special handling of these invalid field-values.
Sending implementations MUST NOT generate Content-Location header
fields that are invalid.
Consuming implementations MAY take steps to recover a usable field-
value from an invalid header field, but SHOULD NOT reject the message
outright, unless this is explicitly desirable behaviour (e.g., the
implementation is a validator). As such, the default handling of
invalid fields is to ignore them.

4. 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 to
use when saving the response payload locally.

4.1. 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>
   quoted-string = <quoted-string, defined in [RFC2616], Section 2.2>
   value        = <value, defined in [RFC2616], Section 3.6>
                ; token | quoted-string
```


Defined in [RFC5987]:

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

Header field values with multiple instances of the same parameter name
are invalid.
Note that due to the rules for implied linear whitespace (Section 2.1
of [RFC2616]), OPTIONAL whitespace can appear between words (token or
quoted-string) and separator characters.
Furthermore note that the format used for ext-value allows specifying a
natural language; this is of limited use for filenames and is likely to
be ignored by recipients.

4.2. Disposition Type

If the disposition type matches "attachment" (case-insensitively), this
indicates that the user agent should prompt the user to save the
response locally, rather than process it normally (as per its media
type).
On the other hand, if it matches "inline" (case-insensitively), this
implies default processing.
Unknown or unhandled disposition types SHOULD be handled by recipients
the same way as "attachment" (see also [RFC2183], Section 2.8).

4.3. 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).
The parameters "filename" and "filename*" differ only in 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]). Many user agent implementations predating this specification do not understand the "filename*" parameter. Therefore, when both "filename" and "filename*" are present in a single header field value, recipients SHOULD pick "filename*" and ignore "filename". This way, senders can avoid special-casing specific user agents by sending both the more expressive "filename*" parameter, and the "filename" parameter as fallback for legacy recipients (see Section 5 for an example).
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 ("\" or "/"),
    recipients SHOULD ignore all but the last path segment. This
    prevents unintentional overwriting of well-known file system
    locations (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 the saved file is
    later opened (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.

   *Recipients are advised to strip or replace character sequences
    that are known to cause confusion both in user interfaces and in
    filenames, such as control characters and leading and trailing
    whitespace.

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

   *Note: Many user agents do not properly handle escape characters
    when using the quoted-string form. Furthermore, some user agents
    erroneously try to perform unescaping of "percent" escapes (see

[Appendix Appendix C.2](#)), and thus might misinterpret filenames
containing the percent character followed by two hex digits.

4.4. Disposition Parameter: Extensions

To enable future extensions, recipients SHOULD ignore unrecognized
parameters (see also [[RFC2183]](#), Section 2.8).

4.5. 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.

5. Examples

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

Content-Disposition: Attachment; filename=example.html

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

  Content-Disposition: INLINE; FILENAME= "an example.html"

Note: this uses the quoted-string form so that the space character can
be included.
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 February 2011, those user agents that do not support the
RFC 5987 encoding ignore "filename*" when it occurs after "filename".

Unfortunately, some user agents that do support RFC 5987 do pick the
"filename" rather than the "filename*" parameter when it occurs first;
it is expected that this situation is going to improve soon.

6. Internationalization Considerations

The "filename*" parameter (Section 4.3), 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.

7. Security Considerations

Using server-supplied information for constructing local filenames
introduces many risks. These are summarized in Section 4.3.
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 Section 5).

8. IANA Considerations

8.1. 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].

8.2. 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 4)

9. Acknowledgements

Thanks to Adam Barth, Rolf Eike Beer, Bjoern Hoehrmann, Alfred Hoenes,
Roar Lauritzsen, Henrik Nordstrom, and Mark Nottingham for their
valuable feedback.

10. References

10.1. Normative References

[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. F., "Character Set and Language Encoding for Hypertext Transfer Protocol (HTTP) Header Field Parameters", RFC 5987, August 2010.
[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.

10.2. 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.
[RFC2388]	Masinter, L., "Returning Values from Forms: multipart/form-data", RFC 2388, August 1998.
[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", STD 66, RFC 3986, 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". The majority of user agents does not implement these, thus
they 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.

Appendix 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.

Appendix C.2. Percent Encoding

Some user agents accept percent encoded ([RFC3986], Section 2.1)
sequences of characters. The character encoding being used for decoding
depends on various factors, including the encoding of the referring
page, the user agent's locale, its configuration, and also the actual
value of the parameter.
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. For
those user agents that do implement this it is difficult to predict
what character encoding they actually expect.

Appendix C.3. Encoding Sniffing

Some user agents inspect the value (which defaults to ISO-8859-1 for
the quoted-string form) 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.

Appendix C.4. Implementations (to be removed by RFC Editor before publication)

Unfortunately, as of February 2011, neither the encoding defined in
RFCs 2231 and 5987, 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:

User Agent	RFC 2231/5987	RFC 2047	Percent Encoding	Encoding Sniffing
Chrome	yes	yes	yes	yes
Firefox	yes (*)	yes	no	yes
Internet Explorer	yes (**)	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 4.3.
(**) Starting with IE9RC, but only implements UTF-8.

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

Note: the issues names in the change log entries for draft-reschke-rfc2183-in-http refer to http://greenbytes.de/tech/webdav/draft-reschke-rfc2183-in-http-issues.html.

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

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

Appendix 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".

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

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

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

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

Appendix D.5. Since draft-ietf-httpbis-content-disp-00

Closed issues:

* http://tools.ietf.org/wg/httpbis/trac/ticket/242: "handling of unknown disposition types"

Slightly updated the notes about the proposed fallback behavior.

Appendix D.6. Since draft-ietf-httpbis-content-disp-01

Various editorial improvements.

Appendix D.7. Since draft-ietf-httpbis-content-disp-02

Closed issues:

* http://tools.ietf.org/wg/httpbis/trac/ticket/244: "state that repeating parameters are invalid"

* http://tools.ietf.org/wg/httpbis/trac/ticket/245: "warn about %xx in filenames being misinterpreted"

* http://tools.ietf.org/wg/httpbis/trac/ticket/246: "mention control chars when talking about postprecessing the filename parameter"

Update Appendix Appendix C.4; Opera 10.63 RC implements the recommended fallback behavior.

Appendix D.8. Since draft-ietf-httpbis-content-disp-03

Closed issues:

* http://tools.ietf.org/wg/httpbis/trac/ticket/252: "'modification-date' *is* implemented in Konq 4.5"

* http://tools.ietf.org/wg/httpbis/trac/ticket/253: "clarify what LWS means for the Content-Disp grammar"

* http://tools.ietf.org/wg/httpbis/trac/ticket/258: "Avoid passive voice in message requirements"

* http://tools.ietf.org/wg/httpbis/trac/ticket/263: "text about historical percent-decoding unclear"

* http://tools.ietf.org/wg/httpbis/trac/ticket/264: "add explanation of language tagging"

* http://tools.ietf.org/wg/httpbis/trac/ticket/265: "Clarify that C-D spec does not apply to multipart upload"

Appendix D.9. Since draft-ietf-httpbis-content-disp-04

Updated implementation information (Chrome 9 implements RFC 5987, IE 9 RC implements it for UTF-8 only).
Clarify who requirements are on, add a section discussing conformance and handling of invalid field values in general.
Closed issues:

* http://tools.ietf.org/wg/httpbis/trac/ticket/272: "Path Separator Characters"

Author's Address

Julian F. Reschke Reschke greenbytes GmbH Hafenweg 16 Muenster, NW 48155 Germany EMail: julian.reschke@greenbytes.de URI: http://greenbytes.de/tech/webdav/