Applications Area Working Group
Internet-Draft
Intended Status: Informational
Expires: June 19, 2015

S. Leonard
Penango, Inc.
December 16, 2014

# The text/markdown Media Type
# draft-ietf-appsawg-text-markdown-04

## Abstract

This document registers the text/markdown media type for use with Markdown, a family of plain text formatting syntaxes that optionally can be converted to formal markup languages such as HTML.

## Status of this Memo

## Copyright Notice

Table of Contents

## 1. Introduction

### 1.1. This Is Markdown! Or: Markup and Its Discontents

In computer systems, textual data is stored and processed using a continuum of techniques. On the one end is plain text: a linear sequence of characters in some character set (code), possibly interrupted by line breaks, page breaks, or other control characters. The repertoire of these control characters (a form of in-band signaling) is necessarily limited, and not particularly extensible. Because they are non-printing, these characters are also hard to enter with standard keyboards.

Markup offers an alternative means to encode this signaling information by overloading certain characters with additional meanings. Therefore, markup languages allow for annotating a document in such a way that annotations are syntactically distinguishable from the printing information. Markup languages are (reasonably) well-specified and tend to follow (mostly) standardized syntax rules. Examples of formal markup languages include SGML, HTML, XML, and LaTeX. Standardized rules lead to interoperability between markup processors, but impose skill requirements on new users that lead to markup languages becoming less accessible to beginners. These rules

also reify "validity": content that does not conform to the rules is treated differently (i.e., is rejected) than content that conforms.

In contrast to formal markup languages, lightweight markup languages use simple syntaxes; they are designed to be easy for humans to enter and understand with basic text editors. Markdown, the subject of this document, began as an /informal/ plain text formatting syntax [MDSYNTAX] and Perl script HTML/XHTML processor [MARKDOWN] targeted at non-technical users using unspecialized tools, such as plain text e-mail clients. [MDSYNTAX] explicitly rejects the notion of validity: there is no such thing as "invalid" Markdown. If the Markdown content does not result in the "right" output (defined as output that the author wants, not output that adheres to some dictated system of rules), the expectation is that the author should continue experimenting by changing the content or the processor to achieve the desired output.

Since its development in 2004 [MARKDOWN], a number of web- and Internet-facing applications have incorporated Markdown into their text entry systems, frequently with custom extensions. Markdown has thus evolved into a kind of Internet meme [INETMEME] as different communities encounter it and adapt the syntax for their specific use cases. Markdown now represents a family of related plain text formatting syntaxes and implementations that, while broadly compatible with humans [HUMANE], are intended to produce different kinds of outputs that push the boundaries of mutual intelligibility between software systems.

To support identifying and conveying Markdown, this document defines a media type and parameters that indicate the author's intent on how to interpret the Markdown. This registration draws particular inspiration from text/troff [RFC4263], which is a plain text formatting syntax for typesetting based on tools from the 1960s ("RUNOFF") and 1970s ("nroff", et. al.). In that sense, Markdown is a kind of troff for modern computing. A companion document [MDMTUSES] provides additional Markdown background and philosophy.

## 1.2. Markdown Is About Writing and Editing

"HTML is a *publishing* format; Markdown is a *writing* format. Thus, Markdown's formatting syntax only addresses issues that can be conveyed in plain text." [MDSYNTAX]

The paradigmatic use case for text/markdown is the Markdown editor: an application that presents Markdown content (which looks like an e-mail or other piece of plain text writing) alongside a published format, so that an author can see results instantaneously and can tweak his or her input in real-time. A significant number of Markdown

editors have adopted "split-screen view" (or "live preview") technology that looks like Figure 1:

```
+-----------------------------------------------------------------------+
| File  Edit  (Cloud Stuff)  (Fork Me on GitHub)  Help                  |
+-----------------------------------------------------------------------+
| [ such-and-such identifier ]                 [ useful statistics]     |
+----------------------------------++-----------------------------------+
| (plain text, with                || (text/html, likely                |
|  syntax highlighting)            ||  rendered to screen)              |
|                                  ||                                   |
|# Introduction                    ||<h1>Introduction</h1>              |
|                                  ||                                   |
|## Markdown Is About Writing and  /|<h2>Markdown Is About Writing and  |
/ Editing                          ||Editing</h2>                       |
|                                  ||                                   |
|> HTML is a *publishing* format;  ||<blockquote><p>HTML is a           |
|> Markdown is a *writing* format. || <em>publishing</em> format;       |
|> Thus, Markdown's formatting     || Markdown is a <em>writing</em>    |
|> syntax only addresses issues    || format. Thus, Markdown's          |
|> that can be conveyed in plain   <> formatting syntax only addresses |
|> text. [MDSYNTAX][]              || issues that can be conveyed in    |
|                                  || plain text. <a href="http://darin/
|The paradigmatic use case for     |/gfireball.net/projects/markdown/sy/
|`text/markdown` is the Markdown   |/ntax#html" title="Markdown: Syntax/
|editor: an application that       |/: HTML">MDSYNTAX</a>               |
|presents Markdown content         ||</p></blockquote>                  |
|...                               ||                                   |
|                                  ||<p>The paradigmatic use case for   |
|[MDSYNTAX]: http://daringfireball./| <code>text/markdown</code> is the|
/net/projects/markdown/syntax#html || Markdown editor: an application   |
|"Markdown: Syntax: HTML"          || that presents Markdown content    |
|                                  || ...</p>                           |
+----------------------------------++-----------------------------------+

 LEGEND: "/" embedded in a vertical line represents a line-continuation
  marker, since a line break is not supposed to occur in that content.
```

Figure 1: Markdown Split-Screen/Live Preview Editor

Users on diverse platforms SHOULD be able to collaborate with their tools of choice, whether those tools are desktop-based (MarkdownPad, MultiMarkdown Composer), browser-based (Dillinger, Markable), integrated widgets (Discourse, GitHub), general-purpose editors (emacs, vi), or plain old "Notepad". Additionally, users SHOULD be able to identify particular areas of Markdown content when the Markdown becomes appreciably large (e.g., book chapters and Internet-Drafts--not just blog posts). Users SHOULD be able to use text/markdown to convey their

works in progress, not just their finished products (for which full-blown markups ranging from text/html to application/pdf are appropriate). This registration facilitates interoperability between these Markdown editors by conveying the syntax of the particular Markdown variant and the desired output format.

### 1.3. Definitions

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].

Since Markdown signifies a family of related formats with varying degrees of formal documentation and implementation, this specification uses the term "variant" to identify such formats.

## 2. Markdown Media Type Registration Application

This section provides the media type registration application for the text/markdown media type (see [RFC6838], Section 5.6).

Type name: text

Subtype name: markdown

Required parameters:

charset: Per Section 4.2.1 of [RFC6838], charset is REQUIRED. There is no default value. [MDSYNTAX] clearly describes Markdown as a writing format; its syntax rules operate on characters (specifically, on punctuation) rather than code points. Neither [MDSYNTAX] nor many popular implementations at the time of this registration actually require or assume any particular encoding. Many Markdown processors will get along just fine by operating on character codes that lie in printable US-ASCII, blissfully oblivious to coded values outside of that range.

Optional parameters:

variant: An optional identifier that serves as a "hint" to the recipient of the specific Markdown variant that the author intended. When omitted, there is no hint; the interpretation is entirely up to the receiver and context. This identifier is plain US-ASCII and case-insensitive. To promote interoperability, identifiers MAY be registered in the registry defined in Section 6. If a receiver does not recognize the variant identifier, the receiver MAY present the identifier to a user to inform him or her of it.

Other parameters MAY be included with the media type. The variant SHOULD define the semantics of such parameters. Additionally, the variant MAY be registered under another media type; this text/markdown registration does not preclude other registrations.

Encoding considerations: Text.

Security considerations:

Markdown interpreted as plain text is relatively harmless. A text editor need only display the text. The editor SHOULD take care to handle control characters appropriately, and to limit the effect of the Markdown to the text editing area itself; malicious Unicode-based Markdown could, for example, surreptitiously change the directionality of the text. An editor for normal text would already take these control characters into consideration, however.

Markdown interpreted as a precursor to other formats, such as HTML, carries all of the security considerations as the target formats. For example, HTML can contain instructions to execute scripts, redirect the user to other webpages, download remote content, and upload personally identifiable information. Markdown also can contain islands of formal markup, such as HTML. These islands of formal markup may be passed as-is, transformed, or ignored (perhaps because the islands are conditional or incompatible) when the Markdown is processed. Since Markdown may have different interpretations depending on the tool and the environment, a better approach is to analyze (and sanitize or block) the output markup, rather than attempting to analyze the Markdown.

Security provides a significant motivator for the output-type parameter. Most Markdown processors emit byte (octet) streams. Without a well-defined means for a Markdown processor to pass metadata onwards, it is perilous for post-processing to assume that the content is always HTML or XHTML. A processor might emit PostScript (application/postscript) content, for example, in which case an HTML sanitizer would fail to excise dangerous instructions.

Interoperability considerations:

Markdown syntaxes are designed to be broadly compatible with humans ("humane"), but not necessarily with each other. Therefore, syntax in one Markdown derivative may be ignored or treated differently in another derivative. The overall effect is a general degradation of the output, proportional to the quantity of syntax-specific Markdown used in the text. When it is desirable to reflect the author's intent in the output, stick with the syntax identified in the syntax parameter.

Published specification: This specification; [MDSYNTAX].

Applications that use this media type:

Markdown conversion tools, Markdown WYSIWYG editors, and plain text editors and viewers; markup processor targets indirectly use Markdown (e.g., web browsers for Markdown converted to HTML).

Fragment identifier considerations:

See Section 4.

Additional information:

Magic number(s): None
File extension(s): .md, .markdown
Macintosh file type code(s):
TEXT. A uniform type identifier (UTI) of "net.daringfireball.markdown", which conforms to "public.plain-text", is RECOMMENDED [MDUTI]. Additionally, implementations SHOULD record syntax and output-type parameters along with the Markdown, such as in extended attributes; however, the exact manner of storage is a local matter.

Person & email address to contact for further information:

Sean Leonard <dev+ietf@seantek.com>

Restrictions on usage: None.

Author/Change controller: Sean Leonard <dev+ietf@seantek.com>

Intended usage: COMMON

Provisional registration? No

## 3.  Optional Parameters

[[NB: OMITTED from this draft. This section may be replaced with Content-Disposition: ... preview-type=...]]

## 4. Fragment Identifiers

Many types of content (such as HTML or PDF) that is output from a Markdown processor will have well-defined fragment identifier semantics associated with the content (such as named anchors or page numbers, respectively). However, the original [MDSYNTAX] neither defines a syntax for naming such content parts, nor associates such

parts with fragment identifiers. Several variants have since defined such content parts, making them suitable for use with fragment identifiers.

## 4.1. General-Purpose Fragment Identifiers

A Markdown fragment identifier is a sequence of characters that identifies some area of the Markdown content. Each Markdown variant can formally define a syntax for such fragment identifiers. (In practice, identifiers that are similar to HTML's anchors are used by many variants, usually by surrounding the identifier with "{#" and "}" and placing the production at the end of a line that comprises particular kinds of content, such as a header, table, or image.) [[NB: citation necessary to PHP Markdown Extra as an exemplary syntax?]]

When encoded in a URI, the production SHALL conform to the fragment production of [RFC3986] (specifically: pchar, "/", and "?" characters). Characters that are outside of that production SHALL be percent-encoded. The character set for percent-encoded octets SHALL be the same as the Markdown content, i.e., identified by the charset parameter or by other contextual means. Variants are free to specify how fragment identifiers are compared. In the absence of a variant-specific rule, fragment identifiers SHOULD be considered case-sensitive, which maintains consistency with HTML. [[NB: citation necessary to HTML4/HTML5?]]

At least the first equals sign "=" SHOULD be percent-encoded to prevent ambiguity as described in the following section.

## 4.2. Parameters

Similar to application/pdf [RFC3778] and text/plain [RFC5147], this registration permits a parameter syntax for fragment identifiers. The syntax is a parameter name, the equals sign "=" (which MUST NOT be percent-encoded), and a parameter value. To the extent that multiple parameters can appear in a fragment production, the parameters SHALL be separated by the ampersand "&" (which MUST NOT be percent-encoded).

The only parameter defined in this registration is "line", which has the same meaning as [RFC5147] (i.e., counting is zero-based). For example: "#line=10" identifies the eleventh line of Markdown input. Implementers should take heed that different environments and character sets may have a wide range of code sequences to divide lines.

Markdown variants are free to define additional parameters.

[[NB: This draft does not import all of text/plain's fragment identifier schemes, mainly because the utility of the other schemes is far from obvious. Implementing line= is not difficult but char= is more difficult since "character" has various meanings that will skew the numbering significantly as the content grows in length; the other integrity check things simply do not seem to be particularly useful.]]

## 5. Example

The following is an example of Markdown as an e-mail attachment:

```
MIME-Version: 1.0
Content-Type: text/markdown; charset=UTF-8; syntax=Original;
 output-type="application/xhtml+xml"
Content-Disposition: attachment; filename=readme.md

Sample HTML 4 Markdown
=============

This is some sample Markdown. [Hooray!][foo]
(Remember that link identifiers are not case-sensitive.)

Bulleted Lists
-------

Here are some bulleted lists...

* One Potato
* Two Potato
* Three Potato

- One Tomato
- Two Tomato
- Three Tomato

More Information
-----------

[.markdown, .md](http://daringfireball.net/projects/markdown/)
has more information.

[fOo]: http://example.com/loc 'Will Not Work with Markdown.pl-1.0.1'
```

## 6. IANA Considerations

IANA is asked to register the media type text/markdown in the Standards tree using the application provided in Section 2 of this

document.

## 6.1. Markdown Variants

IANA is also asked to establish a registry called "Markdown Variants". While the registry is being created in the context of the text/markdown media type, the registry is intended for broad community use, so protocols and systems that do not rely on Internet media types can still tag Markdown content with a common variant identifier. Each entry in this registry shall consist of basic information about the variant:

```
   Identifier
   Name
   Description
   References
   Contact Information
   Expiration Date (if provisional)
```

Variants that have additional media type parameters or fragment identifier considerations SHOULD describe them in detail in the Description field.

While the variant parameter is "plain US-ASCII" (see registration template), the Identifier field (and by implication, all registered identifiers) SHALL conform to the ABNF:

```
  ALPHA [*(ALPHA / DIGIT / "-" / "." / "_" / "~") (ALPHA / DIGIT)]
  [[NB: Be less restrictive, maybe reuse some other common ABNF]]
```

I.e., the identifier MUST start with a letter and MAY contain punctuation in the middle, but not at the end: the last character MUST be alphanumeric. Since the identifier MAY be displayed to a user--particularly in cases where the receiver does not recognize the identifier--the identifier SHOULD be rationally related to the vernacular name of the variant.

The Name, Description, References, and Contact Information fields SHALL be in a Unicode character set (e.g., UTF-8).

## 6.2. Reserved Identifiers

The registry SHALL have the following identifiers RESERVED. No one is allowed to register them (or any case variations of them).

```
   Standard
   Common
   Markdown
```

### 6.3. Standard of Review

Registrations are made on a First-Come, First-Served [RFC5226] basis by anyone with a need to interoperate. While documentation is required, any level of documentation is sufficient; thus, neither Specification Required nor Expert Review are warranted. The checks prescribed by this section can be performed automatically.

All references (including contact information) MUST be verified as functional at the time of the registration.

If a registration is being updated, the contact information MUST either match the prior registration and be verified, or the prior registrant MUST confirm that the updating registrant has authority to update the registration. As a special "escape valve", registrations can be updated with IETF Review [RFC5226]. [[NB: Two purposes: 1) to deal with "harmful" registrations (stale references are not a sufficient justification); 2) to deal with registrations that are IETF registrations, like RFC-related Markdown (but this could be handled by listing the IETF as the contact organization, right?).]] All fields may be updated except the variant identifier, which is permanent: not even case may be changed.

### 6.4. Provisional Registration

Any registrant may make a provisional registration to reserve a variant identifier. Only the variant identifier and contact information fields are required; the rest are optional. Provisional registrations expire after three months, after which time the variant identifier may be reused.

## 7. Security Considerations

See the Security considerations entry in Section 2.

## 8. References

### 8.1. Normative References

[MARKDOWN] Gruber, J., "Daring Fireball: Markdown", December 2004, <http://daringfireball.net/projects/markdown/>.

[MDSYNTAX] Gruber, J., "Daring Fireball: Markdown Syntax Documentation", December 2004, <http://daringfireball.net/projects/markdown/syntax>.

[MDUTI] Gruber, J., "Daring Fireball: Uniform Type Identifier for Markdown", August 2011,

&lt;http://daringfireball.net/linked/2011/08/05/markdown-uti&gt;.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2854] Connolly, D. and L. Masinter, "The 'text/html' Media Type", RFC 2854, June 2000.

[RFC3778] Taft, E., Pravetz, J., Zilles, S., and L. Masinter, "The application/pdf Media Type", RFC 3778, May 2004.

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

[RFC5147] Wilde, E. and M. Duerst, "URI Fragment Identifiers for the text/plain Media Type", RFC 5147, April 2008.

[RFC5226] Narten, T., and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

## 8.2. Informative References

[HUMANE] Atwood, J., "Is HTML a Humane Markup Language?", May 2008, &lt;http://blog.codinghorror.com/is-html-a-humane-markup-language/&gt;.

[INETMEME] Solon, O., "Richard Dawkins on the internet's hijacking of the word 'meme'", June 2013, &lt;http://www.wired.co.uk/news/archive/2013-06/20/richard-dawkins-memes&gt;, &lt;http://www.webcitation.org/6HzDGE9Go&gt;.

[MDMTUSES] Leonard, S., "text/markdown Use Cases", draft-seantek-text-markdown-use-cases-00 (work in progress), October 2014.

[PANDOC]   MacFarlane, J., "Pandoc", 2014, <http://johnmacfarlane.net/pandoc/>.

[RAILFROG] Railfrog Team, "Railfrog", April 2009, <http://railfrog.com/>.

[RFC1468]  Murai, J., Crispin, M., and E. van der Poel, "Japanese Character Encoding for Internet Messages", RFC 1468, June 1993.

[RFC2392]  Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, August 1998.

[RFC3676]  Gellens, R., "The Text/Plain Format and DelSp Parameters", RFC 3676, February 2004.

[RFC4263]  Lilly, B., "Media Subtype Registration for Media Type text/troff", RFC 4263, January 2006.

[FOUNTAIN] Maschwitz, S. and J. August, "Fountain | A markup language for screenwriting.", 2014, <http://fountain.io/>.

[FTSYNTAX] Maschwitz, S. and J. August, "Syntax - Fountain | A markup language for screenwriting.", 1.1, March 2014, <http://fountain.io/syntax>.

## Appendix A.  Change Log

This draft is a continuation from draft-ietf-appsawg-text-markdown-03.txt. These technical changes were made:

1. Removed output-type optional parameter.
2. Renamed syntax optional parameter to variant.
3. Defined variant optional parameter as discussed on mailing list.
4. Removed Section 3 (which may be replaced with Content-Disposition/preview-type in the future).
5. Redid the fragment identifier considerations, simplifying the specification considerably.
6. Discussed the meaning of "variant" in the context of Markdown.
7. Redefined the IANA registry as "Markdown Variants" and expanded its applicability outside of this particular media type.
8. Drastically simplified the registration template.

## Author's Address

Sean Leonard
Penango, Inc.
5900 Wilshire Boulevard
21st Floor
Los Angeles, CA  90036
USA

EMail: dev+ietf@seantek.com
URI:   http://www.penango.com/