

Network Working Group
Internet-Draft
Obsoletes: [3023](#) (if approved)
Updates: [6839](#) (if approved)
Intended status: Standards Track
Expires: August 27, 2014

H. Thompson
University of Edinburgh
C. Lilley
W3C
February 23, 2014

XML Media Types
draft-ietf-appsawg-xml-mediatypes-08

Abstract

This specification standardizes three media types -- application/xml, application/xml-external-parsed-entity, and application/xml-dtd -- for use in exchanging network entities that are related to the Extensible Markup Language (XML) while defining text/xml and text/xml-external-parsed-entity as aliases for the respective application/ types. This specification also standardizes the '+xml' suffix for naming media types outside of these five types when those media types represent XML MIME entities.

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 27, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

Internet-Draft

XML Media Types

February 2014

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Introduction	3
2.	Notational Conventions	4
2.1.	Conformance Keywords	4
2.2.	Characters, Encodings, Charsets	4
2.3.	MIME Entities, XML Entities	4
3.	Encoding Considerations	5
3.1.	XML MIME producers	6
3.2.	XML MIME consumers	6
3.3.	The Byte Order Mark (BOM) and Encoding Conversions	7
4.	XML Media Types	8
4.1.	XML MIME Entities	9
4.2.	Application/xml Registration	10
4.3.	Text/xml Registration	12
4.4.	Application/xml-external-parsed-entity Registration	12
4.5.	Text/xml-external-parsed-entity Registration	13
4.6.	Application/xml-dtd Registration	13
5.	Fragment Identifiers	14
6.	The Base URI	15
7.	XML Versions	15
8.	The '+xml' Naming Convention for XML-Based Media Types	16
8.1.	+xml Structured Syntax Suffix Registration	16
9.	Examples	17
9.1.	UTF-8 Charset	18

9.2.	UTF-16 Charset	18
9.3.	Omitted Charset and 8-bit MIME Entity	19
9.4.	Omitted Charset and 16-bit MIME Entity	19
9.5.	Omitted Charset, no Internal Encoding Declaration	20
9.6.	UTF-16BE Charset	20

9.7.	Non-UTF Charset	20
9.8.	INCONSISTENT EXAMPLE: Conflicting Charset and Internal Encoding Declaration	21
9.9.	INCONSISTENT EXAMPLE: Conflicting Charset and BOM	21
10.	IANA Considerations	22
10.1.	Using '+xml' when Registering XML-based Media Types	22
10.2.	Registration Guidelines for XML-based Media Types Not Using '+xml'	23
11.	Security Considerations	24
12.	References	25
12.1.	Normative References	25
12.2.	Informative References	28
Appendix A.	Why Use the '+xml' Suffix for XML-Based MIME Types?	30
Appendix B.	Core XML specifications	30
Appendix C.	Changes from RFC 3023	31
Appendix D.	Acknowledgements	31
	Authors' Addresses	32

[1.](#) Introduction

The World Wide Web Consortium has issued the Extensible Markup Language (XML) 1.0 [[XML](#)] and Extensible Markup Language (XML) 1.1 [[XML1.1](#)] specifications. To enable the exchange of XML network entities, this specification standardizes three media types -- application/xml, application/xml-external-parsed-entity, and application/xml-dtd and two aliases -- text/xml and text/xml-external-parsed-entity, as well as a naming convention for identifying XML-based MIME media types (using '+xml').

XML has been used as a foundation for other media types, including types in every branch of the IETF media types tree. To facilitate the processing of such types, and in line with the recognition in [[RFC6838](#)] of structured syntax name suffixes, a suffix of '+xml' is described in [Section 8](#). This will allow generic XML-based tools -- browsers, editors, search engines, and other processors -- to work with all XML-based media types.

This specification replaces [\[RFC3023\]](#). Major differences are in the areas of alignment of text/xml and text/xml-external-parsed-entity with application/xml and application/xml-external-parsed-entity respectively, the addition of XPointer and XML Base as fragment identifiers and base URIs, respectively, integration of the XPointer Registry and updating of many references.

[2.](#) Notational Conventions

[2.1.](#) Conformance Keywords

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [\[RFC2119\]](#).

[2.2.](#) Characters, Encodings, Charsets

Both XML (in an XML or Text declaration using the encoding pseudo-attribute) and MIME (in a Content-Type header field using the charset parameter) use a common set of labels [\[IANA-charsets\]](#) to identify the MIME charset (mapping from byte stream to character sequence [\[RFC2978\]](#)).

In this specification we will use the phrases "charset parameter" and "encoding declaration" to refer to whatever MIME charset is specified by a MIME charset parameter or XML encoding declaration respectively. We reserve the phrase "character encoding" (or, when the context makes the intention clear, simply "encoding") for the MIME charset actually used in a particular XML MIME entity.

[UNICODE] defines three "encoding forms", namely UTF-8, UTF-16, and UTF-32. As UTF-8 can only be serialized in one way, the only possible label for UTF-8-encoded documents when serialised into MIME entities is "utf-8". UTF-16 XML documents, however, can be serialised into MIME entities in one of two ways: either big- endian, labelled (optionally) "utf-16" or "utf-16be", or little- endian,

labelled (optionally) "utf-16" or "utf-16le".

UTF-32 has four potential serializations, of which only two (UTF-32BE and UTF-32LE) are given names in in [UNICODE]. Support for the various serializations varies widely, and security concerns about their use have been raised. The use of UTF-32 is NOT RECOMMENDED for XML MIME entities.

[2.3.](#) MIME Entities, XML Entities

As sometimes happens between two communities, both MIME and XML have defined the term entity, with different meanings. [Section 2.4 of \[RFC2045\]](#) says:

"The term 'entity' refers specifically to the MIME-defined header fields and contents of either a message or one of the parts in the body of a multipart entity."

Section 4 of [\[XML\]](#) says:

"An XML document may consist of one or many storage units. These are called entities; they all have content and are all (except for the document entity and the external DTD subset) identified by entity name".

In this specification, "XML MIME entity" is defined as the latter (an XML entity) encapsulated in the former (a MIME entity).

Furthermore, XML provides for the naming and referencing of entities for purposes of inclusion and/or substitution. In this specification "XML-entity declaration/reference/..." is used to avoid confusion when referring to such cases.

[3.](#) Encoding Considerations

The registrations below all address issues around character encoding in the same way, by referencing this section.

As many as three distinct sources of information about character encoding may be present for an XML MIME entity: a charset parameter, a Byte Order Mark (BOM -- see [Section 3.3](#) below) and an XML encoding declaration (see Section 4.3.3 of [\[XML\]](#)). Ensuring consistency among

these sources requires coordination between entity authors and MIME agents (that is, processes which package, transfer, deliver and/or receive MIME entities).

The use of UTF-8, without a BOM, is RECOMMENDED for all XML MIME entities.

Some MIME agents will be what we will call "XML-aware", that is, capable of processing XML MIME entities as XML and detecting the XML encoding declaration (or its absence). All three sources of information about encoding are available to them, and they can be expected to be aware of this spec.

Other MIME agents will not be XML-aware, and thus cannot know anything about the XML encoding declaration. Not only do they lack one of the three sources of information about encoding, they are also less likely to be aware of or responsive to this spec.

Some MIME agents, such as proxies and transcoders, both consume and produce MIME entities.

This mixture of two kinds of agents handling XML MIME entities increases the complexity of the coordination task. The recommendations given below are intended to maximise interoperability in the face of this, by on the one hand mandating consistent production and encouraging maximally robust forms of production, and

on the other specifying recovery strategies to maximize the interoperability of consumers when the production rules are broken.

[3.1](#). XML MIME producers

XML-aware MIME producers SHOULD supply a charset parameter and/or an appropriate BOM with non-UTF-8-encoded XML MIME entities which lack an encoding declaration, and SHOULD remove or correct an encoding declaration which is known to be incorrect (for example, as a result of transcoding).

XML-aware MIME producers MUST supply an XML text declaration at the beginning of non-UNICODE XML external parsed entities which would otherwise begin with the hexadecimal octet sequences 0xFE 0xFF, 0xFF 0xFE or 0xEF 0xBB 0xBF, in order to avoid the mistaken detection of a

BOM.

XML-unaware MIME producers MUST NOT supply a charset parameter with an XML MIME entity unless the entity's character encoding is reliably known. Note that this is particularly relevant for central configuration of web servers, where configuring a default for the charset parameter will almost certainly violate this requirement.

XML MIME producers are RECOMMENDED to provide means for users to control what value, if any, is given to charset parameters for XML MIME entities, for example by giving users control of the configuration of Web server filename-to-Content-Type-header mappings on a file-by-file or suffix basis.

[3.2.](#) XML MIME consumers

For XML MIME consumers, the question of priority arises in cases when the available character encoding information is not consistent. Again, we must distinguish between XML-aware and XML-unaware agents.

When a charset parameter is specified for an XML MIME entity, the normative component of the [\[XML\]](#) specification leaves the question open as to how to determine the encoding with which to attempt to process the entity. This is true independently of whether or not the entity contains in-band encoding information, that is, either a BOM ([Section 3.3](#)) or an XML encoding declaration, or both, or neither. In particular, in the case where there is in-band information and it conflicts with the charset parameter, the [\[XML\]](#) specification does not specify which is authoritative. In its (non-normative) [Appendix F](#) it defers to this specification:

[T]he preferred method of handling conflict should be specified as part of the higher-level protocol used to deliver XML. In particular, please refer to [IETF [RFC 3023](#)] or its successor...

Accordingly, to conform with deployed processors and content and to avoid conflicting with this or other normative specifications, this specification sets the priority as follows:

A BOM ([Section 3.3](#)) is authoritative if it is present in an XML MIME entity;

In the absence of a BOM ([Section 3.3](#)), the charset parameter is authoritative if it is present.

Whenever the above determines a source of encoding information as authoritative, consumers SHOULD process XML MIME entities based on that information.

When MIME producers conform to the requirements stated above ([Section 3.1](#), [Section 3](#)) inconsistencies will not arise---the above statement of priorities only has practical impact in the case of non-conforming XML MIME entities. In the face of inconsistencies, no uniform strategy can deliver the 'right' answer every time: the purpose of specifying one here is to encourage convergence over time, first on the part of consumers, then on the part of producers.

For XML-aware consumers, note that Section 4.3.3 of [[XML](#)] does *not* make it an error for the charset parameter and the XML encoding declaration (or the UTF-8 default in the absence of encoding declaration and BOM) to be inconsistent, although such consumers might choose to issue a warning in this case.

If an XML MIME entity is received where the charset parameter is omitted, no information is being provided about the character encoding by the MIME Content-Type header. XML-aware consumers MUST follow the requirements in section 4.3.3 of [[XML](#)] that directly address this case. XML-unaware MIME consumers SHOULD NOT assume a default encoding in this case.

[3.3](#). The Byte Order Mark (BOM) and Encoding Conversions

Section 4.3.3 of [[XML](#)] specifies that UTF-16 XML MIME entities not labelled as "utf-16le" or "utf16-be" MUST begin with a byte order mark (BOM), U+FEFF, which appears as the hexadecimal octet sequence 0xFE 0xFF (big-endian) or 0xFF 0xFE (little-endian). [[XML](#)] further states that the BOM is an encoding signature, and is not part of either the markup or the character data of the XML document.

UTF-16 to an encoding other than UTF-8 MUST strip the BOM before conversion. Similarly, when converting from another encoding into UTF-16, either without a charset parameter, or labelled "utf-16", the BOM MUST be added unless the original encoding was UTF-8 and a BOM was already present, in which case it MUST be transcoded into the appropriate UTF-16 BOM.

Section 4.3.3 of [\[XML\]](#) also allows for UTF-8 XML MIME entities to begin with a BOM, which appears as the hexadecimal octet sequence 0xEF 0xBB 0xBF. This is likewise defined to be an encoding signature, and not part of either the markup or the character data of the XML document.

Applications that convert XML from UTF-8 to an encoding other than UTF-16 MUST strip the BOM, if present, before conversion. Applications which convert XML into UTF-8 MAY add a BOM.

In addition to the MIME charset "utf-16", [\[RFC2781\]](#) introduces "utf-16le" (little endian) and "utf-16be" (big endian). The BOM is prohibited in MIME entities with these labels. When an XML MIME entity is encoded in "utf-16le" or "utf-16be", it MUST NOT begin with the BOM but SHOULD contain an in-band XML encoding declaration. Conversion from UTF-8 or UTF-16 (unlabelled, or labelled with "utf-16") to "utf-16be" or "utf-16le" MUST strip a BOM if present, and conversion in the other direction MUST (for UTF-16) or MAY (for UTF-8) add the appropriate BOM.

[Appendix F](#) of [\[XML\]](#) also implies the a UTF-32 BOM may be used in conjunction with UTF-32-encoded documents. As noted above, this specification recommends against the use of UTF-32, but if it is used, the same considerations apply with respect to its being a signature, not part of the document, with respect to transcoding into or out of it and with respect to the MIME charsets "utf-32le" and "utf-32be", as for UTF-16. Consumers which do not support UTF-32 SHOULD none-the-less recognise UTF-32 signatures in order to give helpful error messages (instead of treating them as invalid UTF-16).

[4.](#) XML Media Types

Registration information for media types for use with XML MIME entities is described in the sections below, after some relevant background information about XML itself.

[4.1.](#) XML MIME Entities

Within the XML specification, XML MIME entities can be classified into four types. In the XML terminology, they are called "document entities", "external DTD subsets", "external parsed entities", and "external parameter entities". Appropriate usage for the types registered below is as follows:

document entities: The media types `application/xml` or `text/xml`, or a more specific media type (see [Section 8](#)), SHOULD be used.

external DTD subsets: The media type `application/xml-dtd` SHOULD be used. The media types `application/xml` and `text/xml` MUST NOT be used.

external parsed entities: The media types `application/xml-external-parsed-entity` or `text/xml-external-parsed-entity` SHOULD be used. The media types `application/xml` and `text/xml` MUST NOT be used unless the parsed entities are also well-formed "document entities".

external parameter entities: The media type `application/xml-dtd` SHOULD be used. The media types `application/xml` and `text/xml` MUST NOT be used.

Note that [\[RFC3023\]](#) (which this specification obsoletes) recommended the use of `text/xml` and `text/xml-external-parsed-entity` for document entities and external parsed entities, respectively, but described handling of character encoding which differed from common implementation practice. These media types are still commonly used, and this specification aligns the handling of character encoding with industry practice.

Note that [\[RFC2376\]](#) (which is obsolete) allowed `application/xml` and `text/xml` to be used for any of the four types, although in practice it is likely to have been rare.

Neither external DTD subsets nor external parameter entities parse as XML documents, and while some XML document entities may be used as external parsed entities and vice versa, there are many cases where the two are not interchangeable. XML also has unparsed entities, internal parsed entities, and internal parameter entities, but they are not XML MIME entities.

Compared to [\[RFC2376\]](#) or [\[RFC3023\]](#), this specification alters the handling of character encoding of `text/xml` and `text/xml-external-`

parsed-entity, treating them no differently from the respective application/ types. However application/xml and application/xml-

external-parsed-entity are still RECOMMENDED, to avoid possible confusion based on the earlier distinction. The former confusion around the question of default character sets for the two text/ types no longer arises because

[HTTPbis] changes [\[RFC2616\]](#) by removing the ISO-8859-1 default and not defining any default at all;

[RFC6657] updates [\[RFC2046\]](#) to remove the US-ASCII default.

See [Section 3](#) for the now-unified approach to the charset parameter which results.

XML provides a general framework for defining sequences of structured data. It is often appropriate to define new media types that use XML but define a specific application of XML, due to domain-specific display, editing, security considerations or runtime information. Furthermore, such media types may allow only UTF-8 and/or UTF-16 and prohibit other character sets. This specification does not prohibit such media types and in fact expects them to proliferate. However, developers of such media types are RECOMMENDED to use this specification as a basis for their registration. See [Section 8](#) for more detailed recommendations on using the '+xml' suffix for registration of such media types.

An XML document labeled as application/xml or text/xml, or with a '+xml' media type, might contain namespace declarations, stylesheet-linking processing instructions (PIs), schema information, or other declarations that might be used to suggest how the document is to be processed. For example, a document might have the XHTML namespace and a reference to a CSS stylesheet. Such a document might be handled by applications that would use this information to dispatch the document for appropriate processing. [Appendix B](#) lists the core XML specifications which, taken together with [\[XML\]](#) itself, show how to determine an XML document's language-level semantics and suggest how information about its application-level semantics may be locatable.

[4.2.](#) Application/xml Registration

Type name: application

Subtype name: xml

Required parameters: none

Optional parameters: charset

See [Section 3](#).

Encoding considerations: Depending on the character encoding used, XML MIME entities can consist of 7bit, 8bit or binary data [[RFC6838](#)]. For 7-bit transports, 7bit data, for example US-ASCII-encoded data, does not require content-transfer-encoding, but 8bit or binary data, for example UTF-8 or UTF-16 data, MUST be content-transfer-encoded in quoted-printable or base64. For 8-bit clean transport (e.g. 8BITMIME ESMTP [[RFC6152](#)] or NNTP [[RFC3977](#)]), 7bit or 8bit data, for example US-ASCII or UTF-8 data, does not require content-transfer-encoding, but binary data, for example data with a UTF-16 encoding, MUST be content-transfer-encoded in base64. For binary clean transports (e.g. BINARY ESMTP [[RFC3030](#)] or HTTP [[HTTPbis](#)]), no content-transfer-encoding is necessary (or even possible, in the case of HTTP) for 7bit, 8bit or binary data.

Security considerations: See [Section 11](#).

Interoperability considerations: XML has proven to be interoperable across both generic and task-specific applications and for import and export from multiple XML authoring and editing tools. Validating processors provide maximum interoperability. Although non-validating processors may be more efficient, they are not required to handle all features of XML. For further information, see sub-[section 2.9](#) "Standalone Document Declaration" and [section 5](#) "Conformance" of [[XML](#)] .

In practice, character set issues have proved to be the biggest source of interoperability problems. The use of UTF-8, and careful attention to the guidelines set out in [Section 3](#), are the best way to avoid such problems.

Published specification: Extensible Markup Language (XML) 1.0 (Fifth Edition) [[XML](#)] or subsequent editions or versions thereof.

Applications that use this media type: XML is device-, platform-, and vendor-neutral and is supported by generic and task-specific applications and a wide range of generic XML tools (editors, parsers, Web agents, ...).

Additional information:

Magic number(s): None.

Although no byte sequences can be counted on to always be present, XML MIME entities in ASCII-compatible character sets (including UTF-8) often begin with hexadecimal 3C 3F 78 6D 6C ("<?xml"), and those in UTF-16 often begin with hexadecimal FE

Thompson & Lilley

Expires August 27, 2014

[Page 11]

Internet-Draft

XML Media Types

February 2014

FF 00 3C 00 3F 00 78 00 6D 00 6C or FF FE 3C 00 3F 00 78 00 6D 00 6C 00 (the Byte Order Mark (BOM) followed by "<?xml"). For more information, see [Appendix F](#) of [[XML](#)].

File extension(s): .xml

Macintosh File Type Code(s): "TEXT"

Base URI: See [Section 6](#)

Person and email address for further information: See Authors' Addresses section

Intended usage: COMMON

Author: See Authors' Addresses section

Change controller: The XML specification is a work product of the World Wide Web Consortium's XML Core Working Group. The W3C has change control over this specification.

[4.3.](#) Text/xml Registration

The registration information for text/xml is in all respects the same as that given for application/xml above ([Section 4.2](#)).

[4.4.](#) Application/xml-external-parsed-entity Registration

Type name: application

Subtype name: xml-external-parsed-entity

Required parameters: none

Optional parameters: charset

See [Section 3](#).

Encoding considerations: Same as for application/xml ([Section 4.2](#)).

Security considerations: See [Section 11](#).

Interoperability considerations: XML external parsed entities are as interoperable as XML documents, though they have a less tightly constrained structure and therefore need to be referenced by XML documents for proper handling by XML processors. Similarly, XML documents cannot be reliably used as external parsed entities because external parsed entities are prohibited from having

standalone document declarations or DTDs. Identifying XML external parsed entities with their own content type enhances interoperability of both XML documents and XML external parsed entities.

Published specification: Same as for application/xml ([Section 4.2](#)).

Applications which use this media type: Same as for application/xml ([Section 4.2](#)).

Additional information:

Magic number(s): Same as for application/xml ([Section 4.2](#)).

File extension(s): .xml or .ent

Macintosh File Type Code(s): "TEXT"

Base URI: See [Section 6](#)

Person and email address for further information: See Authors' Addresses section.

Intended usage: COMMON

Author: See Authors' Addresses section.

Change controller: The XML specification is a work product of the World Wide Web Consortium's XML Core Working Group. The W3C has change control over this specification.

[4.5.](#) Text/xml-external-parsed-entity Registration

The registration information for text/xml-external-parsed-entity is in all respects the same as that given for application/xml-external-parsed-entity above ([Section 4.4](#)).

[4.6.](#) Application/xml-dtd Registration

Type name: application

Subtype name: xml-dtd

Required parameters: none

Optional parameters: charset

See [Section 3](#).

Encoding considerations: Same as for application/xml ([Section 4.2](#)).

Security considerations: See [Section 11](#).

Interoperability considerations: XML DTDs have proven to be interoperable by DTD authoring tools and XML validators, among others.

Published specification: Same as for application/xml ([Section 4.2](#)).

Applications which use this media type: DTD authoring tools handle

external DTD subsets as well as external parameter entities. XML validators may also access external DTD subsets and external parameter entities.

Additional information:

Magic number(s): Same as for application/xml ([Section 4.2](#)).

File extension(s): .dtd or .mod

Macintosh File Type Code(s): "TEXT"

Person and email address for further information: See Authors' Addresses section.

Intended usage: COMMON

Author: See Authors' Addresses section.

Change controller: The XML specification is a work product of the World Wide Web Consortium's XML Core Working Group. The W3C has change control over this specification.

[5.](#) Fragment Identifiers

Uniform Resource Identifiers (URIs) can contain fragment identifiers (see [Section 3.5 of \[RFC3986\]](#)). Specifying the syntax and semantics of fragment identifiers is devolved by [\[RFC3986\]](#) to the appropriate media type registration.

The syntax and semantics of fragment identifiers for the XML media types defined in this specification are based on the [\[XPointerFramework\]](#) W3C Recommendation. It allows simple names, and more complex constructions based on named schemes. When the syntax of a fragment identifier part of any URI or IRI ([\[RFC3987\]](#)) with a retrieved media type governed by this specification conforms to the syntax specified in [\[XPointerFramework\]](#), conforming applications MUST

interpret such fragment identifiers as designating whatever is specified by the [\[XPointerFramework\]](#) together with any other specifications governing the XPointer schemes used in those identifiers which the applications support. Conforming applications

MUST support the 'element' scheme as defined in [\[XPointerElement\]](#), but need not support other schemes.

If an XPointer error is reported in the attempt to process the part, this specification does not define an interpretation for the part.

A registry of XPointer schemes [\[XPtrReg\]](#) is maintained at the W3C. Document authors SHOULD NOT use unregistered schemes. Scheme authors SHOULD register their schemes ([\[XPtrRegPolicy\]](#) describes requirements and procedures for doing so).

See [Section 10.2](#) for additional requirements which apply when an XML-based media type follows the naming convention '+xml'.

If [\[XPointerFramework\]](#) and [\[XPointerElement\]](#) are inappropriate for some XML-based media type, it SHOULD NOT follow the naming convention '+xml'.

When a URI has a fragment identifier, it is encoded by a limited subset of the repertoire of US-ASCII characters, see [\[XPointerFramework\]](#) for details..

[6.](#) The Base URI

An XML MIME entity of type application/xml, text/xml, application/xml-external-parsed-entity or text/xml-external-parsed-entity MAY use the xml:base attribute, as described in [\[XMLBase\]](#), to embed a base URI in that entity for use in resolving relative URI references (see [Section 5.1 of \[RFC3986\]](#)).

Note that the base URI itself might be embedded in a different MIME entity, since the default value for the xml:base attribute can be specified in an external DTD subset or external parameter entity. Since conforming XML processors need not always read and process external entities, the effect of such an external default is uncertain and therefore its use is NOT RECOMMENDED.

[7.](#) XML Versions

application/xml, application/xml-external-parsed-entity, and application/xml-dtd, text/xml and text/xml-external-parsed-entity are to be used with [\[XML\]](#). In all examples herein where version="1.0" is shown, it is understood that version="1.1" might also appear, providing the content does indeed conform to [\[XML1.1\]](#).

The normative requirement of this specification upon XML documents and processors is to follow the requirements of [\[XML\]](#), section 4.3.3. Except for minor clarifications, that section is substantially identical from the first edition to the current (5th) edition of XML 1.0, and for XML 1.1 1st or 2nd edition [\[XML1.1\]](#). Therefore, references herein to [\[XML\]](#) may be interpreted as referencing any existing version or edition of XML, or any subsequent edition or version which makes no incompatible changes to that section.

Specifications and recommendations based on or referring to this RFC SHOULD indicate any limitations on the particular versions or editions of XML to be used.

[8.](#) The '+xml' Naming Convention for XML-Based Media Types

This section supersedes the earlier registration of the '+xml' suffix [\[RFC6839\]](#).

This specification recommends the use of the '+xml' naming convention for identifying XML-based media types, in line with the recognition in [\[RFC6838\]](#) of structured syntax name suffixes. This allows the use of generic XML processors and technologies on a wide variety of different XML document types at a minimum cost, using existing frameworks for media type registration.

See [Section 10](#) for guidance on when and how to register a '+xml'-based media subtype, and on registering a media subtype for XML but not using '+xml'.

[8.1.](#) +xml Structured Syntax Suffix Registration

Name: Extensible Markup Language (XML)

+suffix: +xml

Reference: This specification

Encoding considerations: Same as [Section 4.2](#).

Fragment identifier considerations: Registrations which use this '+xml' convention MUST also make reference to RFC XXXX, specifically [Section 5](#), in specifying fragment identifier syntax and semantics, and they MAY restrict the syntax to a specified subset of schemes, except that they MUST NOT disallow barenames or 'element' scheme pointers. They MAY further require support for other registered schemes. They also MAY add additional syntax (which MUST NOT overlap with [\[XPointerFramework\]](#) syntax) together

with associated semantics, and MAY add additional semantics for

Internet-Draft

XML Media Types

February 2014

barename XPointers which, as provided for in [Section 5](#), will only apply when this specification does not define an interpretation.

In practice these constraints imply that for a fragment identifier addressed to an instance of a specific "xxx/yyy+xml" type, there are three cases:

For fragment identifiers matching the syntax defined in [\[XPointerFramework\]](#), where the fragment identifier resolves per the rules specified there, then process as specified there;

For fragment identifiers matching the syntax defined in [\[XPointerFramework\]](#), where the fragment identifier does not resolve per the rules specified there, then process as specified in "xxx/yyy+xml";

For fragment identifiers not matching the syntax defined in [\[XPointerFramework\]](#), then process as specified in "xxx/yyy+xml". A fragment identifier of the form "xywh=160,120,320,240", as defined in [\[MediaFrag\]](#), which might be used in a URI for an XML-encoded image, would fall in this category.

Interoperability considerations: Same as [Section 4.2](#). See above, and also [Section 3](#), for guidelines on the use of the 'charset' parameter.

Security considerations: See [Section 11](#).

Contact: See Authors' Addresses section.

Author: See Authors' Addresses section.

Change controller: The XML specification is a work product of the World Wide Web Consortium's XML Core Working Group. The W3C has change control over this specification.

[9.](#) Examples

This section is non-normative. In particular, note that all [\[RFC2119\]](#) language herein reproduces or summarizes the consequences of normative statements already made above, and has no independent normative force, and accordingly does not appear in uppercase.

The examples below give the MIME Content-type header, including the charset parameter, if present and the XML declaration or Text declaration (which includes the encoding declaration) inside the XML MIME entity. For UTF-16 examples, the Byte Order Mark character appropriately UTF-16-encoded is denoted as "{BOM}", and the XML or Text declaration is assumed to come at the beginning of the XML MIME entity, immediately following the encoded BOM. Note that other MIME headers may be present, and the XML MIME entity will normally contain other data in addition to the XML declaration; the examples focus on the Content-type header and the encoding declaration for clarity.

Although they show a content type of 'application/xml', all the examples below apply to all five media types declared above in [Section 4](#), as well as to any media types declared using the '+xml' convention (with the exception of the examples involving the charset parameter for any such media types which do not enable its use). See the XML MIME entities table ([Section 4.1](#), Paragraph 1) for discussion of which types are appropriate for which varieties of XML MIME entity.

[9.1.](#) UTF-8 Charset

Content-Type: application/xml; charset=utf-8

```
<?xml version="1.0" encoding="utf-8"?>
```

or

```
<?xml version="1.0"?>
```

UTF-8 is the recommended encoding for use with all the media types defined in this specification. Since the charset parameter is provided and there is no overriding BOM, conformant MIME and XML

processors must treat the enclosed entity as UTF-8 encoded.

If sent using a 7-bit transport (e.g. SMTP [[RFC5321](#)]), in general, a UTF-8 XML MIME entity must use a content-transfer-encoding of either quoted-printable or base64. For an 8-bit clean transport (e.g. 8BITMIME ESMTP or NNTP), or a binary clean transport (e.g. BINARY ESMTP or HTTP), no content-transfer-encoding is necessary (or even possible, in the case of HTTP).

[9.2.](#) UTF-16 Charset

Content-Type: application/xml; charset=utf-16

```
{BOM}<?xml version="1.0" encoding="utf-16"?>
```

or

```
{BOM}<?xml version="1.0"?>
```

For the three application/ media types defined above, if sent using a 7-bit transport (e.g. SMTP) or an 8-bit clean transport (e.g. 8BITMIME ESMTP or NNTP), the XML MIME entity must be encoded in quoted-printable or base64; for a binary clean transport (e.g. BINARY ESMTP or HTTP), no content-transfer-encoding is necessary (or even possible, in the case of HTTP).

As described in [[RFC2781](#)], the UTF-16 family must not be used with media types under the top-level type "text" except over HTTP or HTTPS (see section A.2 of HTTP [[HTTPbis](#)] for details). Hence one of the two text/ media types defined above can be used with this example only when the XML MIME entity is transmitted via HTTP or HTTPS, which use a MIME-like mechanism and are binary-clean protocols, hence do not perform CR and LF transformations and allow NUL octets. Since HTTP is binary clean, no content-transfer-encoding is necessary (or even possible).

[9.3.](#) Omitted Charset and 8-bit MIME Entity

Content-Type: application/xml

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

Since the charset parameter is not provided in the Content-Type header and there is no overriding BOM, conformant XML processors must treat the "iso-8859-1" encoding as authoritative. Conformant XML-unaware MIME processors should make no assumptions about the character encoding of the XML MIME entity.

[9.4.](#) Omitted Charset and 16-bit MIME Entity

Content-Type: application/xml

```
{BOM}<?xml version="1.0" encoding="utf-16"?>
```

or

```
{BOM}<?xml version="1.0"?>
```

This example shows a 16-bit MIME entity with no charset parameter. However since there is a BOM conformant processors must treat the entity as UTF-16-encoded.

Omitting the charset parameter is not recommended in conjunction with media types under the top-level type "application" when used with transports other than HTTP or HTTPS. Media types under the top-level type "text" should not be used for 16-bit MIME with transports other than HTTP or HTTPS (see discussion above ([Section 9.2](#), Paragraph 7)).

[9.5.](#) Omitted Charset, no Internal Encoding Declaration

Content-Type: application/xml

```
<?xml version='1.0'?>
```

In this example, the charset parameter has been omitted, there is no internal encoding declaration, and there is no BOM. Since there is no BOM or charset parameter, the XML processor follows the requirements in [section 4.3.3](#), and optionally applies the mechanism described in [Appendix F](#) (which is non-normative) of [\[XML\]](#) to determine an encoding of UTF-8. Although the XML MIME entity does not contain an encoding declaration, provided the encoding actually

is UTF-8, this is a conforming XML MIME entity.

A conformant XML-unaware MIME processor should make no assumptions about the character encoding of the XML MIME entity.

See [Section 9.1](#) for transport-related issues for UTF-8 XML MIME entities.

[9.6.](#) UTF-16BE Charset

Content-Type: application/xml; charset=utf-16be

```
<?xml version='1.0' encoding='utf-16be'?>
```

Observe that, as required for this encoding, there is no BOM. Since the charset parameter is provided and there is no overriding BOM, conformant MIME and XML processors must treat the enclosed entity as UTF-16BE encoded.

See also the additional considerations in the UTF-16 example ([Section 9.2](#)) above.

[9.7.](#) Non-UTF Charset

Content-Type: application/xml; charset=iso-2022-kr

```
<?xml version="1.0" encoding="iso-2022-kr"?>
```

This example shows the use of a non-UTF character encoding (in this case Hangul, but this example is intended to cover all non-UTF-family character encodings). Since the charset parameter is provided and there is no overriding BOM, conformant processors must treat the enclosed entity as encoded per [RFC 1557](#).

Since ISO-2022-KR [[RFC1557](#)] has been defined to use only 7 bits of data, no content-transfer-encoding is necessary with any transport: for character sets needing 8 or more bits, considerations such as those discussed above ([Section 9.1](#), [Section 9.2](#)) would apply.

[9.8.](#) INCONSISTENT EXAMPLE: Conflicting Charset and Internal Encoding

Declaration

Content-Type: application/xml; charset=iso-8859-1

```
<?xml version="1.0" encoding="utf-8"?>
```

Although the charset parameter is provided in the Content-Type header and there is no BOM and the charset parameter differs from the XML encoding declaration, conformant MIME and XML processors will interoperate. Since the charset parameter is authoritative in the absence of a BOM, conformant processors will treat the enclosed entity as iso-8859-1 encoded. That is, the "UTF-8" encoding declaration will be ignored.

Conformant processors generating XML MIME entities must not label conflicting character encoding information between the MIME Content-Type and the XML declaration unless they have definitive information about the actual encoding, for example as a result of systematic transcoding. In particular, the addition by servers of an explicit, site-wide charset parameter default has frequently lead to interoperability problems for XML documents.

[9.9.](#) INCONSISTENT EXAMPLE: Conflicting Charset and BOM

Content-Type: application/xml; charset=iso-8859-1

```
{BOM}<?xml version="1.0"?>
```

Although the charset parameter is provided in the Content-Type header, there is a BOM, so MIME and XML processors may not interoperate. Since the BOM parameter is authoritative for conformant XML processors, they will treat the enclosed entity as UTF-16-encoded. That is, the "iso-8859-1" charset parameter will be ignored. XML-unaware MIME processors on the other hand may be unaware of the BOM and so treat the entity as encoded in iso-8859-1.

Conformant processors generating XML MIME entities must not label conflicting character encoding information between the MIME Content-Type and an entity-initial BOM.

[10.](#) IANA Considerations

As described in [Section 8](#), this specification updates the [\[RFC6839\]](#) registration for XML-based MIME types (the '+xml' types).

[10.1](#). Using '+xml' when Registering XML-based Media Types

When a new media type is introduced for an XML-based format, the name of the media type SHOULD end with '+xml' unless generic XML processing is in some way inappropriate for documents of the new type. This convention will allow applications that can process XML generically to detect that the MIME entity is supposed to be an XML document, verify this assumption by invoking some XML processor, and then process the XML document accordingly. Applications may check for types that represent XML MIME entities by comparing the last four characters of the subtype to the string '+xml'. (However note that 4 of the 5 media types defined in this specification -- text/xml, application/xml, text/xml-external-parsed-entity, and application/xml-external-parsed-entity -- also represent XML MIME entities while not ending with '+xml'.)

NOTE: [Section 5.3.2](#) of HTTPbis [\[HTTPbis\]](#) does not support any form of Accept header which will match only '+xml' types. In particular, Accept headers of the form "Accept: */*+xml" are not allowed, and so this header MUST NOT be used for this purpose.

Media types following the naming convention '+xml' SHOULD introduce the charset parameter for consistency, since XML-generic processing applies the same program for any such media type. However, there are some cases that the charset parameter need not be introduced. For example:

When an XML-based media type is restricted to UTF-8, it is not necessary to introduce the charset parameter. UTF-8 is the default for XML.

When an XML-based media type is restricted to UTF-8 and UTF-16, it might not be unreasonable to omit the charset parameter. Neither UTF-8 nor UTF-16 require XML encoding declarations.

XML generic processing is not always appropriate for XML-based media types. For example, authors of some such media types may wish that the types remain entirely opaque except to applications that are specifically designed to deal with that media type. By NOT following

the naming convention '+xml', such media types can avoid XML-generic processing. Since generic processing will be useful in many cases, however -- including in some situations that are difficult to predict ahead of time -- the '+xml' convention is to be preferred unless there is some particularly compelling reason not to.

The registration process for specific '+xml' media types is described in [[RFC6838](#)]. The registrar for the IETF tree will encourage new XML-based media type registrations in the IETF tree to follow this guideline. Registrars for other trees SHOULD follow this convention in order to ensure maximum interoperability of their XML-based documents. Only media subtypes that represent XML MIME entities are allowed to register with a '+xml' suffix.

In addition to the changes described above, the change controller has been changed to be the World Wide Web Consortium (W3C).

10.2. Registration Guidelines for XML-based Media Types Not Using '+xml'

Registrations for new XML-based media types which do not use the '+xml' suffix SHOULD, in specifying the charset parameter and encoding considerations, define them as: "Same as [charset parameter / encoding considerations] of application/xml as specified in RFC XXXX."

Enabling the charset parameter is RECOMMENDED, since this information can be used by XML processors to determine authoritatively the character encoding of the XML MIME entity in the absence of a BOM. If there are some reasons not to follow this advice, they SHOULD be included as part of the registration. As shown above, two such reasons are "UTF-8 only" or "UTF-8 or UTF-16 only".

These registrations SHOULD specify that the XML-based media type being registered has all of the security considerations described in RFC XXXX plus any additional considerations specific to that media type.

These registrations SHOULD also make reference to RFC XXXX in specifying magic numbers, base URIs, and use of the BOM.

These registrations MAY reference the application/xml registration in RFC XXXX in specifying interoperability and fragment identifier considerations, if these considerations are not overridden by issues specific to that media type.

Internet-Draft

XML Media Types

February 2014

11. Security Considerations

XML MIME entities contain information which may be parsed and further processed by the recipient. These entities may contain, and recipients may permit, explicit system level commands to be executed while processing the data. To the extent that a recipient application executes arbitrary command strings from within XML MIME entities, they may be at risk.

In general, any information stored outside of the direct control of the user -- including CSS style sheets, XSL transformations, XML-entity declarations, and DTDs -- can be a source of insecurity, by either obvious or subtle means. For example, a tiny "whiteout attack" modification made to a "master" style sheet could make words in critical locations disappear in user documents, without directly modifying the user document or the stylesheet it references. Thus, the security of any XML document is vitally dependent on all of the documents recursively referenced by that document.

The XML-entity lists and DTDs for XHTML 1.0 [[XHTML](#)], for instance, are likely to be a commonly used set of information. Many developers will use and trust them, few of whom will know much about the level of security on the W3C's servers, or on any similarly trusted repository.

The simplest attack involves adding declarations that break validation. Adding extraneous declarations to a list of character XML-entities can effectively "break the contract" used by documents. A tiny change that produces a fatal error in a DTD could halt XML processing on a large scale. Extraneous declarations are fairly obvious, but more sophisticated tricks, like changing attributes from being optional to required, can be difficult to track down. Perhaps the most dangerous option available to attackers, when external DTD subsets or external parameter entities or other externally-specified defaulting is involved, is redefining default values for attributes: e.g. if developers have relied on defaulted attributes for security, a relatively small change might expose enormous quantities of information.

Apart from the structural possibilities, another option, "XML-entity spoofing," can be used to insert text into documents, vandalizing and perhaps conveying an unintended message. Because XML permits

multiple XML-entity declarations, and the first declaration takes precedence, it is possible to insert malicious content where an XML-entity reference is used, such as by inserting the full text of Winnie the Pooh in place of every occurrence of —.

Security considerations will vary by domain of use. For example, XML medical records will have much more stringent privacy and security considerations than XML library metadata. Similarly, use of XML as a parameter marshalling syntax necessitates a case by case security review.

XML may also have some of the same security concerns as plain text. Like plain text, XML can contain escape sequences that, when displayed, have the potential to change the display processor environment in ways that adversely affect subsequent operations. Possible effects include, but are not limited to, locking the keyboard, changing display parameters so subsequent displayed text is unreadable, or even changing display parameters to deliberately obscure or distort subsequent displayed material so that its meaning is lost or altered. Display processors SHOULD either filter such material from displayed text or else make sure to reset all important settings after a given display operation is complete.

Some terminal devices have keys whose output, when pressed, can be changed by sending the display processor a character sequence. If this is possible the display of a text object containing such character sequences could reprogram keys to perform some illicit or dangerous action when the key is subsequently pressed by the user. In some cases not only can keys be programmed, they can be triggered remotely, making it possible for a text display operation to directly perform some unwanted action. As such, the ability to program keys SHOULD be blocked either by filtering or by disabling the ability to program keys entirely.

Note that it is also possible to construct XML documents that make use of what XML terms "[XML-]entity references" to construct repeated expansions of text. Recursive expansions are prohibited by [\[XML\]](#) and XML processors are required to detect them. However, even non-recursive expansions may cause problems with the finite computing resources of computers, if they are performed many times. For

example, consider the case where XML-entity A consists of 100 copies of XML-entity B, which in turn consists of 100 copies of XML-entity C, and so on.

[12.](#) References

[12.1.](#) Normative References

- [HTTPbis] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", [draft-ietf-httpbis-p1-messaging-25](#) (work in progress), November 2013.

[IANA-charsets]

IANA, "Character Sets Registry", 2013,
<[http://www.iana.org/assignments/character-sets/
character-sets.xhtml](http://www.iana.org/assignments/character-sets/character-sets.xhtml)>.

- [RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", [RFC 2045](#), November 1996.
- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", [RFC 2046](#), November 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2781] Hoffman, P. and F. Yergeau, "UTF-16, an encoding of ISO 10646", [RFC 2781](#), February 2000.
- [RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", [RFC 2978](#), October 2000.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax.", [RFC 3986](#), January 2005.
- [RFC3987] Dueerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", [RFC 3987](#), July 2005.

[RFC6657] Melnikov, A. and J. Reschke, "Update to MIME regarding "charset" Parameter Handling in Textual Media Types", [RFC 6657](#), July 2012, <<http://www.rfc-editor.org/rfc/rfc6657.txt>>.

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

[RFC6839] Hansen, T. and A. Melnikov, "Additional Media Type Structured Syntax Suffixes", [RFC 6839](#), January 2013.

[UNICODE] The Unicode Consortium, "The Unicode Standard, Version 6.3.0", 2013, <<http://www.unicode.org/versions/Unicode6.3.0/>>.

Defined by: The Unicode Standard, Version 6.3 (Mountain View, CA: The Unicode Consortium, 2013. ISBN 978-1-936213-08-5)

[XML1.1] Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., Yergeau, F., and J. Cowan, "Extensible Markup Language (XML) 1.1 (Second Edition)", W3C Recommendation REC-xml, September 2006, <<http://www.w3.org/TR/2006/REC-xml11-20060816/>>.

Latest version available at [2].

[XMLBase] Marsh, J. and R. Tobin, "XML Base (Second Edition)", W3C Recommendation REC-xmlbase-20090128, January 2009, <<http://www.w3.org/TR/2009/REC-xmlbase-20090128/>>.

Latest version available at [3].

[XML] Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation REC-xml, November 2008, <<http://www.w3.org/TR/2008/REC-xml-20081126/>>.

Latest version available at [1].

[XPointerElement]

Grosso, P., Maler, E., Marsh, J., and N. Walsh, "XPointer element() Scheme", W3C Recommendation REC-XPointer-Element, March 2003,
<<http://www.w3.org/TR/2003/REC-xptr-element-20030325/>>.

Latest version available at [4].

[XPointerFramework]

Grosso, P., Maler, E., Marsh, J., and N. Walsh, "XPointer Framework", W3C Recommendation REC-XPointer-Framework, March 2003,
<<http://www.w3.org/TR/2003/REC-xptr-framework-20030325/>>.

Latest version available at [5].

[XPtrRegPolicy]

Hazael-Massieux, D., "XPointer Scheme Name Registry Policy", 2005,
<<http://www.w3.org/2005/04/xpointer-policy.html>>.

[XPtrReg]

Hazael-Massieux, D., "XPointer Registry", 2005,
<<http://www.w3.org/2005/04/xpointer-schemes/>>.

12.2. Informative References

[ASCII]

American National Standards Institute, "Coded Character Set -- 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[AWWW]

Jacobs, I. and N. Walsh, "Architecture of the World Wide Web, Volume One", W3C Recommendation REC-webarch-20041215, December 2004,
<<http://www.w3.org/TR/2004/REC-webarch-20041215/>>.

Latest version available at [8].

[FYN]

Mendelsohn, N., "The Self-Describing Web", W3C TAG Finding

selfDescribingDocuments-2009-02-07, February 2009,
<[http://www.w3.org/2001/tag/doc/
selfDescribingDocuments-2009-02-07.html](http://www.w3.org/2001/tag/doc/selfDescribingDocuments-2009-02-07.html)>.

Latest version available at [9]

- [InfoSet] Cowan, J. and R. Tobin, "XML Information Set (Second Edition)", W3C Recommendation REC-xml-infoSet-20040204, February 2004,
<<http://www.w3.org/TR/2005/REC-xml-id-20050909/>>.

Latest version available at [11].

- [MediaFragments] Troncy, R., Mannens, E., Pfeiffer, S., and D. Van Deursen, "Media Fragments URI 1.0 (basic)", W3C Recommendation media-frag, September 2012,
<<http://www.w3.org/TR/2012/REC-media-frag-20120925/>>.

Latest version available at [6].

- [RFC1557] Choi, U., Chon, K., and H. Park, "Korean Character Encoding for Internet Messages", [RFC 1557](#), December 1993.
- [RFC2376] Whitehead, E. and M. Murata, "XML Media Types", [RFC 2376](#), July 1998.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Nielsen, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.
- [RFC3023] Murata, M., St-Laurent, S., and D. Kohn, "XML Media Types", [RFC 3023](#), January 2001.

- [RFC3030] Vaudreuil, G., "SMTP Service Extensions for Transmission of Large and Binary MIME Messages", [RFC 3030](#), 2000.
- [RFC3977] Feather, B., "Network News Transfer Protocol", [RFC 3977](#), October 2006.
- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", [RFC 5321](#),

October 2008.

[RFC6152] Klensin, J., Freed, N., Rose, M., and D. Crocker, "SMTP Service Extension for 8-bit MIME Transport", [RFC 6152](#), March 2011.

[TAGMIME] Bray, T., Ed., "Internet Media Type registration, consistency of use", April 2004,
<<http://www.w3.org/2001/tag/2004/0430-mime>>.

[XHTML] Pemberton, S. and et al, "XHTML 1.0: The Extensible HyperText Markup Language", W3C Recommendation xhtml1, December 1999,
<<http://www.w3.org/TR/2000/REC-xhtml1-20000126/>>.

Latest version available at [7].

[XMLModel] Grosso, P. and J. Kosek, "Associating Schemas with XML documents 1.0 (Third Edition)", W3C Group Note NOTE-xml-model-20121009, October 2012,
<<http://www.w3.org/TR/2012/NOTE-xml-model-20121009/>>.

Latest version available at [13].

[XMLNS10] Bray, T., Hollander, D., Layman, A., Tobin, R., and H. Thompson, "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation REC-xml-names-20091208, December 2009,
<<http://www.w3.org/TR/2009/REC-xml-names-20091208/>>.

Latest version available at [12].

[XMLNS11] Bray, T., Hollander, D., Layman, A., and R. Tobin, "Namespaces in XML 1.1 (Second Edition)", W3C Recommendation REC-xml-names11-20060816, August 2006,
<<http://www.w3.org/TR/2006/REC-xml-names11-20060816/>>.

Latest version available at [14].

[XMLSS] Clark, J., Pieters, S., and H. Thompson, "Associating Style Sheets with XML documents 1.0 (Second Edition)", W3C Recommendation REC-xml-stylesheet-20101028, October 2010, <<http://www.w3.org/TR/2010/REC-xml-stylesheet-20101028/>>.

Latest version available at [15].

[XMLid] Marsh, J., Veillard, D., and N. Walsh, "xml:id Version 1.0", W3C Recommendation REC-xml-id-20050909, September 2005, <<http://www.w3.org/TR/2005/REC-xml-id-20050909/>>.

Latest version available at [10].

[Appendix A](#). Why Use the '+xml' Suffix for XML-Based MIME Types?

[RFC3023] contains a detailed discussion of the (at the time) novel use of a suffix, a practice which has since become widespread. Interested parties are referred to [\[RFC3023\]](#), [Appendix A](#).

The registration process for new '+xml' media types is described in [\[RFC6838\]](#)

[Appendix B](#). Core XML specifications

The following specifications each articulate key aspects of XML document semantics:

Namespaces in XML 1.0 [\[XMLNS10\]](#)/Namespaces in XML 1.1 [\[XMLNS11\]](#)

XML Information Set [\[Infoset\]](#)

xml:id [\[XMLid\]](#)

XML Base [\[XMLBase\]](#)

Associating Style Sheets with XML documents [\[XMLSS\]](#)

Associating Schemas with XML documents [\[XMLModel\]](#)

The W3C Technical Architecture group has produced two documents which are also relevant:

The Self-Describing Web [\[FYN\]](#) discusses the overall principles of how document semantics are determined on the Web.

Architecture of the World Wide Web, Volume One [\[Awww\]](#), [section 4.5.4](#), discusses the specific role of XML Namespace documents in this process.

Internet-Draft

XML Media Types

February 2014

[Appendix C](#). Changes from [RFC 3023](#)

There are numerous and significant differences between this specification and [\[RFC3023\]](#), which it obsoletes. This appendix summarizes the major differences only.

XPointer ([\[XPointerFramework\]](#) and [\[XPointerElement\]](#)) has been added as fragment identifier syntax for all the XML media types, and the XPointer Registry ([\[XPtrReg\]](#)) mentioned

[\[XMLBase\]](#) has been added as a mechanism for specifying base URIs

The language regarding character sets was updated to correspond to the W3C TAG finding Internet Media Type registration, consistency of use [\[TAGMIME\]](#)

Priority is now given to a Byte Order Mark (BOM) if present

Many references are updated, and the existence of XML 1.1 and relevance of this specification to it acknowledged

A number of justifications and contextualizations which were appropriate when XML was new have been removed, including the whole of the original [Appendix A](#)

Making BOMs authoritative is in principle a backwards-incompatibility. In practice serious interoperability issues already exist when BOMs are used. Making BOMs authoritative, in conjunction with the deprecation of the UTF-32 encoding form and the requirement to include an XML encoding declaration in certain cases ([Section 3.1](#)), is intended to improve in-practice interoperability as much as possible.

[Appendix D](#). Acknowledgements

MURATA Makoto (FAMILY Given) and Alexey Melnikov made early and important contributions to the effort to revise [\[RFC3023\]](#).

This specification reflects the input of numerous participants to the ietf-xml-mime@imc.org, xml-mime@ietf.org and apps-discuss@ietf.org mailing lists, though any errors are the responsibility of the authors. Special thanks to:

Mark Baker, James Clark, Dan Connolly, Martin Duerst, Ned Freed, Yaron Goland, Bjoern Hoehrmann, Rick Jelliffe, Murray S. Kucherawy, Larry Masinter, David Megginson, S. Moonesamy, Keith Moore, Chris Newman, Gavin Nicol, Julian Reschke, Marshall Rose, Jim Whitehead,

Thompson & Lilley

Expires August 27, 2014

[Page 31]

Internet-Draft

XML Media Types

February 2014

Erik Wilde and participants of the XML activity and the TAG at the W3C.

Jim Whitehead and Simon St.Laurent were editors of [[RFC2376](#)] and [[RFC3023](#)], respectively.

Authors' Addresses

Henry S. Thompson
University of Edinburgh

Email: ht@inf.ed.ac.uk
URI: <http://www.ltg.ed.ac.uk/~ht/>

Chris Lilley
World Wide Web Consortium
2004, Route des Lucioles - B.P. 93 06902
Sophia Antipolis Cedex
France

Email: chris@w3.org
URI: <http://www.w3.org/People/chris/>

Thompson & Lilley

Expires August 27, 2014

[Page 32]