

Network Working Group  
Internet-Draft  
Updates: [4287](#) (if approved)  
Intended status: Informational  
Expires: January 30, 2014

E. Wilde  
EMC  
July 29, 2013

Profile Support for the Atom Syndication Format  
draft-wilde-atom-profile-02

## Abstract

The Atom syndication format is a generic XML format for representing collections. Profiles are one way how Atom feeds can indicate that they support specific extensions. To make this support visible on the media type level, this specification adds an optional "profile" media type parameter to the Atom media type. This allows profiles to become visible at the media type level, so that servers as well as clients can indicate support for specific Atom profiles in conversations, for example when communicating via HTTP. This specification updates [RFC 4287](#) by adding the "profile" media type parameter to the application/atom+xml media type registration.

## Note to Readers

This draft should be discussed on the atom-syntax mailing list [[10](#)].

Online access to all versions and files is available on github [[11](#)].

## 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 January 30, 2014.

## Copyright Notice

Internet-Draft

Atom Profiles

July 2013

Copyright (c) 2013 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 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.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">2.</a>	Examples . . . . .	<a href="#">3</a>
<a href="#">2.1.</a>	Profiles for Alternatives . . . . .	<a href="#">4</a>
<a href="#">2.2.</a>	Profiles for Specializations . . . . .	<a href="#">4</a>
<a href="#">2.3.</a>	Profile URI for AtomPub . . . . .	<a href="#">5</a>
<a href="#">3.</a>	Profile Parameter Definition . . . . .	<a href="#">5</a>
<a href="#">4.</a>	IANA Considerations . . . . .	<a href="#">6</a>
<a href="#">4.1.</a>	Atom Media Type application/atom+xml . . . . .	<a href="#">6</a>
<a href="#">5.</a>	Implementation Status . . . . .	<a href="#">6</a>
<a href="#">6.</a>	Security Considerations . . . . .	<a href="#">7</a>
<a href="#">7.</a>	Open Issues . . . . .	<a href="#">7</a>
<a href="#">8.</a>	Change Log . . . . .	<a href="#">7</a>
<a href="#">8.1.</a>	From -01 to -02 . . . . .	<a href="#">7</a>
<a href="#">8.2.</a>	From -00 to -01 . . . . .	<a href="#">8</a>
<a href="#">9.</a>	References . . . . .	<a href="#">8</a>
<a href="#">9.1.</a>	Normative References . . . . .	<a href="#">8</a>
<a href="#">9.2.</a>	Non-Normative References . . . . .	<a href="#">8</a>
<a href="#">Appendix A.</a>	Acknowledgements . . . . .	<a href="#">9</a>
	Author's Address . . . . .	<a href="#">9</a>

## 1. Introduction

The Atom Syndication Format "is an XML-based document format that describes lists of related information known as 'feeds'. Feeds are composed of a number of items, known as 'entries', each with an extensible set of attached metadata. For example, each entry has a title." [\[1\]](#)

Profiles "can be described as additional semantics that can be used to process a resource representation, such as constraints, conventions, extensions, or any other aspects that do not alter the basic media type semantics. A profile MUST NOT change the semantics of the resource representation when processed without profile knowledge, so that clients both with and without knowledge of a profiled resource can safely use the same representation." [\[2\]](#)

Profiles are identified by URI, and their use can be indicated for a representation by adding a link with the registered "profile" link relation type, linking to the profile URI. While this is sufficient to represent the fact that a certain representation is using a profile, it does not make that fact visible outside of this representation. Ideally, peers communicating their media type, for example when communicating via Hypertext Transfer Protocol (HTTP) [\[5\]](#), should be able to indicate the support of certain profiles through the media type identifier itself, without changing the base media type.

Because Atom supports generic links through its `<link/>` element, "profile" links can be easily added to a feed, indicating that this feed does adhere to a certain profile. However, on the media type level, this feed would still be labeled as `application/atom+xml`, making the profile invisible on that level and thus not allowing it to be used in interactions such as content negotiation in HTTP.

This specification adds a "profile" media type parameter to the `application/atom+xml` media type, thereby making it possible for

profiles to be exposed at the media type level. Apart from adding that one media type parameter, this specification does not change anything about the Atom format itself, or its media type registration.

## [2.](#) Examples

Adding a "profile" parameter to the Atom media type adds visibility of profiles at the media type level, for example when alternative profiles are supported by a service. It might also help to further "specialize" a media type in environments where such a

"specialization" is useful. Two examples are intended to illustrate these two scenarios.

### [2.1.](#) Profiles for Alternatives

For example, when linking to feeds of media-oriented services, it would be possible to expose two feeds, one using MediaRSS, and the other one using Podcasts. Both formats roughly cover the same functionality as media-oriented feed-based extensions, but by having the ability to expose their capabilities at the media type level, HTTP mechanisms and conversations can be used to distinguish between these formats.

In some cases it may be possible to support more than one profile, and then it is up for the service to decide whether these should be exposed in one representation (which can be exposed by linking to multiple profiles from the resource representation and/or in the media type parameter), or whether there should be two representations, one for each profile. This decision will probably depend on implementation complexity, the trade-off between navigation complexity (two representations with one profile each) and processing complexity, and also the size of the profile data, because in particular in the case of overlapping profiles, there might be many redundancies.

Thus, which way to go for multiple profiles is not a question that has one correct answer; it depends on the profiles, and on the services that are built around them.

## [2.2.](#) Profiles for Specializations

Feed-based services may provide additional features in feeds that are represented using Atom's extension mechanisms. These additional features might be useful only for those clients that support them, and otherwise might add volume to a feed that is of no value to general consumers. In such a scenario, specialized clients might also request their specialized features via profile media type parameters, and will then get the feed being "enriched" with the additional features. If clients do not request such a profile or request one that is not known to the server, the server responds with a generic feed, still allowing them to treat the feed as a generic feed (with no additional features being represented).

Whether services respond with profiles by default or only for specific requests about a profile is a matter of policy, and will be influenced by factors such as the added volume when adding profile data, and the question whether profiles should only be exposed to those that specifically ask for them. Since profiles are not allowed

to change the semantics of the media type itself, such a decision can depend on the trade-off being a matter of expressivity, and not whether it will break clients under some circumstances.

## [2.3.](#) Profile URI for AtomPub

The Atom Publishing Protocol (AtomPub) [6] builds on Atom and defines additional interactions with feeds, such as the ability to POST an entry to a collection URI as a request to create a new entry in that collection. AtomPub uses Atom's media type for representing feeds and entries (and introduces its own media type for representing category and service documents, but these are not relevant for this discussion).

When requesting a collection URI from an AtomPub server, clients will GET a feed document with no indication that the server supports AtomPub. Clients are supposed to have knowledge about AtomPub support, so that they know whether POST requests to the collection URI might succeed. It is possible that clients send an OPTIONS request to the collection URI to find out about the allowed methods, but this requires an additional roundtrip, and since the AtomPub spec does not explicitly mention OPTIONS, it may be the case that

implementations do not generally support this discovery mechanism.

To make AtomPub support of a collection explicit in a feed document, the profile URI `urn:ietf:rfc:5023` is suggested. When including this profile URI in a feed, a server indicates AtomPub support:

```
<?xml version="1.0" ?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <link rel="profile" href="urn:ietf:rfc:5023">
```

When used with the profile parameter of the Atom media type, this profile URI MAY be used to indicate that the resource is advertising AtomPub support. It should be noted that AtomPub servers are not required to use the AtomPub profile URI in any way (because it is not a part of the AtomPub specification), but that supporting it may make it easier for clients to discover the AtomPub capabilities of available resources.

### [3.](#) Profile Parameter Definition

The profile parameter for the `application/atom+xml` media type allows one or more profile URIs to be specified. These profile URIs have the identifier semantics defined in [\[2\]](#), and when appearing as media type parameter, they have the same semantics as if they had been associated with the resource URI through other means, such as using one or more `<link profile="" href=""/>` elements as children of the

`<feed>` element.

As a general rule, media type parameters must be quoted unless they are tokens. For the "profile" media type parameter defined here, this means that it must be quoted. It contains a non-empty list of space-separated URIs (the profile URIs).

```
profile-param = "profile=" profile-value
profile-value = <"> profile-URI 0*( 1*SP profile-URI ) <">
profile-URI   = URI
```

The "URI" in the above grammar refers to the "URI" as defined in Section 3 of [\[3\]](#)

### [4.](#) IANA Considerations

This specification updates an existing media type according to the registry mechanism described in [7].

#### [4.1.](#) Atom Media Type application/atom+xml

The Internet media type for Atom (application/atom+xml) should be updated by adding the following optional media type parameter:

##### [4.1.1.](#) Optional Parameters

profile: This parameter indicates that one or more profiles are used in the feed, according to the definition of profiles in [2]. The parameter syntax is specified in [Section 3](#) of RFC XXXX

## [5.](#) Implementation Status

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

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC 6982](#) [8]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC 6982](#), "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

...

## [6.](#) Security Considerations

There are no known security considerations for adding this optional media type parameter to the application/atom+xml media type.

## [7.](#) Open Issues

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

- o Monitor how the proposal for a "Profile URI Registry" [[9](#)] is coming along. If it is successful, then the proposed AtomPub Profile URI [Section 2.3](#) should be included in the IANA Considerations [Section 4](#).

## [8.](#) Change Log

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

### [8.1.](#) From -01 to -02

- o Added "Implementation Status" section ([Section 5](#))."
- o Added example and suggested URI for an AtomPub Profile ([Section 2.3](#))
- o Changed IANA section to only request adding a "profile" media type parameter (instead of providing a complete media type registration template).
- o Added "Open Issues" section ([Section 7](#)) and reminder to check the progress of the "Profile URI Registry" draft.
- o Updating "Implementation Status" section to refer to [RFC 6982](#) [[8](#)].
- o Adding "Security Considerations" section ([Section 6](#))

### [8.2.](#) From -00 to -01



- o Fixed typos.
- o Removed the requirement to percent-encode URIs in the profile parameter.
- o Added example for media type specialization.

## 9. References

### 9.1. Normative References

- [1] Nottingham, M., Ed. and R. Sayre, Ed., "The Atom Syndication Format", [RFC 4287](#), December 2005.
- [2] Wilde, E., "The 'profile' Link Relation Type", [RFC 6906](#), March 2013.
- [3] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), January 2005.
- [4] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", [RFC 3023](#), January 2001.

### 9.2. Non-Normative References

- [5] 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.
- [6] Gregorio, J. and B. de h0ra, "The Atom Publishing Protocol", [RFC 5023](#), October 2007.
- [7] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", [BCP 13](#), [RFC 6838](#), January 2013.
- [8] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", [RFC 6982](#), July 2013.
- [9] Lanthaler, M., "The IETF Profile URI Registry", [draft-lanthaler-profile-registry-02](#) (work in progress), June 2013.

URIs

[10] <<http://www.imc.org/atom-syntax/>>

[11] <<https://github.com/dret/I-D/tree/master/atom-profile>>

## [Appendix A](#). Acknowledgements

Thanks for comments and suggestions provided by Markus Lanthaler and Peter Rushforth.

## Author's Address

Erik Wilde  
EMC  
6801 Koll Center Parkway  
Pleasanton, CA 94566  
U.S.A.

Phone: +1-925-6006244  
Email: [erik.wilde@emc.com](mailto:erik.wilde@emc.com)  
URI: <http://dret.net/netdret/>

Wilde

Expires January 30, 2014

[Page 9]