Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 20, 2013

E. Wilde
EMC Corporation
October 17, 2012

The 'profile' Link Relation Type
draft-wilde-profile-link-04

Abstract

This specification defines the 'profile' link relation type that allows resource representations to indicate that they are following one or more profiles. A profile is defined to not alter the semantics of the resource representation itself, but to allow clients to learn about additional semantics (constraints, conventions, extensions) that are associated with the resource representation, in addition to those defined by the media type and possibly other mechanisms.

Editorial Note (to be removed by RFC Editor)

Please discuss this draft on the apps-discuss@ietf.org mailing list.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

One of the foundations of the Internet and Web Architecture is the fact that resource representations communicated through protocols such as SMTP or HTTP are labeled with a 'media type', which allows a client to understand at run time what 'type' of resource representation it is handling. Sometimes, it would be useful for servers and clients to include additional information about the nature of the resource, so that a client understanding this additional information could react in a way specific to that specialization of the resource, where the specialization can be about constraints, conventions, extensions, or any other aspects that do not alter the basic media type semantics. HTML 4 [HTML401] has such a mechanism built into the language, which is the 'profile' attribute of the 'head' element. This mechanism, however, is specific to HTML alone, and at the time of writing it seems as if HTML 5 will drop support for this mechanism entirely.

RFC 5988 [RFC5988] "defines a framework for typed links that is not specific to a particular serialization or application. It does so by redefining the link relation registry established by Atom to have a broader domain, and adding to it the relations that are defined by HTML."

This specification registers a 'profile' link relation type according to the rules of RFC 5988 [RFC5988]. This link relation type is independent of the context in which it is used (however, the representation must support typed links for this mechanism to work) and does not constrain in any way the target of the linked URI. In fact, for the purpose of this specification, the target URI does not necessarily have to identify a dereferencable resource (or even use a dereferencable URI scheme), and clients can treat the occurrence of a specific URI in the same way as an XML namespace URI and invoke specific behavior based on the assumption that a specific profile target URI signals that a resource representation follows a specific profile. Note that at the same time, it is possible for profile target URIs to use dereferencable URIs and use a representation (which is outside the scope of this specification) which represents the information about the profile in a human- or machine-readable way.

As one example, consider the case of podcasts, a specific kind of feed using additional fields for media-related metadata. Using a 'profile' link, it would be easily possible for clients to understand that a specific feed is supposed to be a podcast feed, and that it may contain entries using podcast-specific fields. This may allow a client to behave differently when handling such a feed (such as rendering a podcast-specific UI), even when the current set of

entries in the feed may not contain any podcast entries.

2. Terminology

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 RFC 2119 [RFC2119].

3. Profiles

The concept of a profile has no strict definition on the Internet or on the Web. For the purpose of this specification, a profile 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. While this specification associates profiles with resource representations, creators and users of profiles MAY define and manage them in a way that they can be used across media types and thus could be associated with a resource, independent of its representations (i.e., using the same profile URI for different media types). However, such a design is outside of the scope of this specification, and clients SHOULD treat profiles as being associated with a resource representation.

Profiles can be combined, meaning that a single resource representation can conform to zero or any number of profiles. Depending on the profile support of clients, it is possible that the same resource representation, when linked to a number of profiles, can be processed with different sets of processing rules, based on the profile support of the clients.

Profiles are identified by URI, but as with for example XML namespace URIs, the URI in this case only serves as an identifier, meaning that the presence of a specific URI has to be sufficient for a client to assert that a resource representation conforms to a profile. Clients thus SHOULD treat profile URIs as identifiers and not as links, but profiles MAY be defined in a way that the URIs do identify retrievable profile description and thus can be accessed by clients by dereferencing the profile URI. For profiles intended for use in environments where clients may encounter unknown profile URIs, profile maintainers SHOULD consider to make the profile URI dereferencable and provide useful documentation at that URI. The design and representation of such profile descriptions, however, is

outside the scope of this specification.

3.1. Profiles and Media Types

A media type defines both the semantics and the serialization of a specific type of content. In many cases, media types have some extensibility or openness built-in, so that specific instances of the media type can layer additional semantics on top of the media type's foundations. In this case, a profile is the appropriate mechanism to signal that the original semantics and processing model of the media type still applies, but that an additional processing model can be used to extract additional semantics. This is in contrast to a new media type, which instead of just adding processing rules and semantics, in most cases defines a complete set of processing rules and semantics. As an example, XHTML is not a profile of XML but a new media type because it introduces a complete new perspective of the underlying XML structures, and from the XHTML point of view, exposing the raw XML is not all that useful for clients. However, hCard (see Section 5.1) is a profile of (X)HTML because it adds processing rules that allow a client to extract additional semantics from a representation, without changing any of the processing rules and semantics of (X)HTML itself. While the line between a media type and a profile might not always be easy to draw, the intention of profiles is not to replace media types, but to add a more lightweight and runtime-capable mechanism that allows servers and clients to be more explicit in how a specific instance of a media type represents concepts that are not defined by the media type itself, but by additional conventions (the profile processing rules and semantics).

The idea of profiles is that they allow instances to clearly identify what kind of mechanism they are using for expressing additional semantics, should they follow a well-defined framework for doing so (see Section 5 for examples). While this allows servers and clients to represent the use of profiles, it does not make the profile information visible outside of the representation itself, if the representation is using embedded typed links. For newly defined media types that may be used with profiles, it is therefore recommended that they SHOULD define a media type parameter called "profile", and specify that this media type parameter follows the semantics of a profile as laid out in this document. This way, clients can use this media type parameter to request a certain profile when interacting, for example, with an HTTP server and setting the Accept header. Representations using a "profile" media type parameter still SHOULD include that value in the representation using the "profile" link relation, since the media type label of a representation can easily get lost when it is taken out of its conversational context.

Since a representation can link to more than one profile, the same has to be possible for the corresponding media type parameter (if a media type defines such a parameter). Media types defining a "profile" parameter SHOULD define it as a whitespace-separated list of profile URIs.

3.2. Profile Context

Profile links convey information about the use of profiles for a media type. If they are used within a media type, they apply to the context specified by that media type, which means that for example profile links in the head element of an HTML document apply to the document as a whole. The context of a profile extends to the scope of where it is being used, which means that profiles used in profile media type parameters (as described in Section 3.1) or used in HTTP Link headers extend to the scope of the protocol in which they are being used.

4. IANA Considerations

The link relation type below will be registered by IANA per Section 6.2.1 of RFC 5988 [RFC5988]:

Relation Name: profile

Description: Identifying that a resource representation conforms to a certain profile, without affecting the non-profile semantics of the resource representation.

Reference: [[This document]]

Notes: Profile URIs are primarily intended to be used as identifiers, and thus clients SHOULD NOT indiscriminately access profile URIs.

5. Examples

This section lists some examples of profiles that already are defined today (and thus could be readily used with a 'profile' link), and of some potential additional examples. Since so far, profiles have been mostly limited to HTML (because of the support of profiles in HTML), the two examples of existing profiles are HTML profiles, and the two hypothetical examples are non-HTML examples.

5.1. hCard

The hCard profile uses http://microformats.org/profile/hcard as its defining URI and is essentially a mechanism how vCard [RFC6350] information can be embedded in an HTML page using the mechanisms provided by microformats. It is thus a good example for how profiles might on the one hand define a model-based extension of the original media type (in this case adding vCard fields), and how they also have to define specific ways of how that model extension then is represented in the media type (in this case, using microformats). Alternatively, it would be possible to represent vCard information through the mechanisms of RDFa or microdata, but since these would be different conventions that a client would need to follow to extract the vCard data, they would be identified by different profiles.

5.2. Dublin Core

Dublin Core metadata identified by the profile http://dublincore.org/documents/2008/08/04/dc-html/ can be used to embed Dublin Core metadata in an HRML page. In contrast to hCard, which is using microformats as its foundation, the Dublin Core profile defines its own way of embedding metadata into HTML, and does so by using HTML <link> elements. The interesting difference to hCard is that Dublin Core not only defines metadata to be embedded in HTML, it also allows links to be added as metadata, in which case the profile not just describes additional data to be found within the representation, but also allows the representation to be linked to additional resources.

5.3. Podcasts

Podcasts are an extension of feed formats, and define a substantial set of additional attributes to reflect the fact that the resources in podcast feeds are time-based media formats such as audio and video. While there is no profile URI for podcasts, the current definition (maintained by Apple) at http://www.apple.com/itunes/podcasts/specs.html could serve as such a URI, or it could by updated to include such a URI. Podcasts are feeds with special behavior, and while it is possible to follow a podcast feed using a generic feed reader, a podcast-aware feed reader will be able to extract additional information from the feed, and thus can implement more sophisticated services or present a more sophisticated UI for podcast feeds. The Apple page referenced above describes the implementation of one such specialized podcast feed reader, Apple iTunes.

6. Security Considerations

The 'profile' relation type is not known to introduce any new security issues not already discussed in RFC 5988 [RFC5988] for generic use of Web linking mechanisms.

7. Change Log

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

7.1. From -03 to -04

- o Changed category from "Standard" to "Informational".
- o Minor textual changes.

7.2. From -02 to -03

- o Removed the AtomPub example, which seemed to cause more confusion than clarification.
- o Minor textual changes.

7.3. From -01 to -02

- o Feedback directed to apps-discuss@ietf.org
- o Improved explanation of difference between media types and profiles in "Profiles and Media Types" (Section 3.1).
- o Added section on "Profiles and Media Types" (Section 3.1).
- o Added section on "Profile Context" (Section 3.2).

7.4. From -00 to -01

- o Updated security considerations.
- o Made it clear that profiles are about resource representations, and not about resources.
- o Added examples section (Section 3.1) with four examples (Dublin Core, HCard, AtomPub, and Podcasts).
- o Minor textual changes.

8. References

8.1. Normative References

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

[RFC5988] Nottingham, M., "Web Linking", RFC 5988, October 2010.

8.2. Informative References

[HTML401] Hors, A., Raggett, D., and I. Jacobs, "HTML 4.01 Specification", World Wide Web Consortium Recommendation REC-html401-19991224, December 1999, <http://www.w3.org/TR/1999/REC-html401-19991224>.

[RFC6350] Perreault, S., "vCard Format Specification", RFC 6350, August 2011.

Appendix A. Acknowledgements

Thanks for comments and suggestions provided by Erlend Hamnaberg, Markus Lanthaler, Simon Mayer, Mark Nottingham, Julian Reschke, James Snell, and Tim Williams.

Author's Address

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

Phone: +1-925-6006244
Email: erik.wilde@emc.com
URI: http://dret.net/netdret/