Network Working Group
Request for Comments: 4316
Category: Experimental

J. Reschke
greenbytes

December 2005

Datatypes for Web Distributed Authoring and Versioning (WebDAV) Properties

Status of This Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This specification extends the Web Distributed Authoring and Versioning Protocol (WebDAV) to support datatyping. Protocol elements are defined to let clients and servers specify the datatype, and to instruct the WebDAV method PROPFIND to return datatype information.

Table of Contents

1. Introduction

This specification builds on the infrastructure provided by the Web Distributed Authoring and Versioning (WebDAV) Protocol, adding support for data-typed properties.

Although servers must support XML content in property values, it may be desirable to persist values as scalar values when possible and to expose the data's type when the property value is returned to the client. The client is free to ignore this information, but it may be able to take advantage of it when modifying a property.

On the other hand, when setting new properties, it can be desirable to pass datatype information along with the value. A server can take advantage of this information to optimize storage and to perform additional parsing (for instance, of dates). Servers that support searching can also take advantage of known datatypes when doing comparisons and sorting.

The following potential datatyping-related features were deliberately considered out of scope:

- getting "schema" information for classes of resources (set of "required" properties, their types, display information),
- definition of a set of mandatory property types,
- discovery of supported property types,
- extensions to PROPPATCH that would allow updates to parts of a (structured) property.

2. Notational Conventions

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

The term "property element" refers to the XML element that identifies a particular property, for instance,

```
<getcontentlength xmlns="DAV:" />
```

The term "prop element" is used for the WebDAV "prop" element as defined in Section 12.11 of [RFC2518].

The XML representation of schema components uses a vocabulary identified by the namespace name "http://www.w3.org/2001/XMLSchema".

For brevity, the text and examples in this specification use the prefix "xs:" to stand for this namespace; in practice, any prefix can be used. "XML Schema Part 1: Structures" ([XS1]) also defines several attributes for direct use in any XML documents. These attributes are in a different namespace named "http://www.w3.org/2001/XMLSchema-instance". For brevity, the text and examples in this specification use the prefix "xsi:" to stand for this latter namespace; in practice, any prefix can be used.

3. Overview

Although WebDAV property types can be anything that can be marshaled as content of an XML element, in many cases they actually are simple types like integers, booleans, or dates. "XML Schema Part 2: Datatypes" [XS2] defines a set of simple types that can be used as a basis for supplying type information to attributes.

Datatype information is represented using the attribute "type" from the XML Schema namespace "http://www.w3.org/2001/XMLSchema-instance". In XML Schema, datatypes are qualified names, and the XML Schema recommendation defines a set of built-in datatypes (Section 3 of [XS2]), defined in the namespace "http://www.w3.org/2001/XMLSchema".

To avoid unnecessary verbosity, datatype information should only be supplied if it adds usable information to the protocol. In particular, type information is not required for live properties defined in WebDAV [RFC2518] and for properties of type "xs:string".

A server may implement any combination of datatypes, both from the XML Schema recommendation and possibly from other namespaces.

Note that a particular property can be typed for a number of reasons:

- The property is a live property with server-defined semantics and value space.
- The property may have been set using a non-WebDAV protocol that the server understands in addition to WebDAV.
- The type may have been specified in an extended PROPPATCH method as defined in Section 4.

4. Changes for PROPPATCH Method

If the property element has an XML attribute named "xsi:type", the server may use this information to select an optimized representation for storing the property value. For instance, by specifying a type as "xs:boolean", the client declares the property value to be of type boolean (as defined in [XS2]). The server may choose any suitable internal format for persisting this property, and in particular is allowed to fail the request if the format given does not fit the format defined for this type.

The server should indicate successful detection and parsing of the typed value by setting the xsi:type attribute on the property element in the response body (this implies that it should return a MULTISTATUS status code and a <multistatus> response body).

4.1. Example of Successful PROPPATCH

```
>>Request

PROPPATCH /bar.html HTTP/1.1
Host: example.org
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:set>
    <D:prop>
      <Z:released xsi:type="xs:boolean">false</Z:released>
    </D:prop>
  </D:set>
</D:propertyupdate>

>>Response

HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:Z="http://ns.example.org/standards/z39.50">
```

```
    <D:response>
      <D:href>http://example.org/bar.html</D:href>
      <D:propstat>
        <D:prop><Z:released xsi:type="xs:boolean" /></D:prop>
        <D:status>HTTP/1.1 200 OK</D:status>
      </D:propstat>
    </D:response>
  </D:multistatus>
```

In this case, the xsi:type attribute on the element "Z:released" indicates that the server indeed has understood the submitted data type information.

4.2. Example of Failed PROPPATCH

```
>>Request

PROPPATCH /bar.html HTTP/1.1
Host: example.org
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:set>
    <D:prop>
      <Z:released xsi:type="xs:boolean">t</Z:released>
    </D:prop>
  </D:set>
</D:propertyupdate>

>>Response

HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:"
   xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:response>
    <D:href>http://example.org/bar.html</D:href>
    <D:propstat>
      <D:prop><Z:released/></D:prop>
      <D:status>HTTP/1.1 422 Unprocessable Entity</D:status>
```

```
        <D:responsedescription>
          Does not parse as xs:boolean
        </D:responsedescription>
      </D:propstat>
    </D:response>
  </D:multistatus>
```

In this case, the request failed because the supplied value "t" is not a valid representation for a boolean value.

Note that similar error conditions can occur in the standard WebDAV protocol even though no datatype was specified: for instance, when a client tries to set a live property for which only a certain value space is allowed.

4.3. Example of Successful PROPPATCH Where Type Information Was Not Preserved

```
>>Request

PROPPATCH /bar.html HTTP/1.1
Host: example.org
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:set>
    <D:prop>
      <Z:released xsi:type="Z:custom">t</Z:released>
    </D:prop>
  </D:set>
</D:propertyupdate>

>>Response

HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:response>
    <D:href>http://example.org/bar.html</D:href>
```

```
      <D:propstat>
        <D:prop><Z:released/></D:prop>
        <D:status>HTTP/1.1 200 OK</D:status>
      </D:propstat>
    </D:response>
  </D:multistatus>
```

In this case, the request succeeded, but the server did not know how to handle the datatype "Z:custom". Therefore, no datatype information was returned in the response body.

5. Changes for PROPFIND Method

PROPFIND is extended to return the datatype information for properties by adding "xsi:type" attributes to the property elements unless one of the following conditions is met:

- The datatype MUST be different from "xs:string" (because this can be considered the default datatype).
- The property's datatype MUST NOT be defined in [RFC2518] (because these types are already well-defined).

5.1. Example of PROPFIND/prop

```
>>Request

PROPFIND /bar.html HTTP/1.1
Host: example.org
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propfind xmlns:D="DAV:"
  xmlns:Z="http://ns.example.org/standards/z39.50">
  <D:prop>
    <D:getcontenttype/>
    <Z:released/>
  </D:prop>
</D:propfind>

>>Response

HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

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

```
<D:multistatus xmlns:D="DAV:"
   xmlns:Z="http://ns.example.org/standards/z39.50"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <D:response>
    <D:href>http://example.org/bar.html</D:href>
    <D:propstat>
      <D:prop>
        <D:getcontenttype>text/html</D:getcontenttype>
        <Z:released xsi:type="xs:boolean">1</Z:released>
      </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
  </D:response>
</D:multistatus>
```

This example shows that the property value "true" is returned with the correct datatype information and that the server chose one of the two possible representations defined in XML Schema. It also shows that datatype information is not returned for "D:getcontenttype", as this property's datatype is already defined in [RFC2518].

6. Changes for Other Methods

Servers that support other methods using the DAV:multistatus response format (such as the REPORT method defined in [RFC3253], Section 3.6) SHOULD apply the same extensions as defined in Section 5.

7. Compatibility Considerations

This part of this specification does not introduce any new protocol elements, nor does it change the informal WebDAV DTD. It merely specifies additional server semantics for the case where clients submit additional datatype information in an attribute on the property element (previously undefined), and adds an additional attribute on property elements upon PROPFIND.

Clients not aware of datatype handling should not supply the "xsi: type" attribute on property elements (after all, this attribute belongs to the XML Schema-Instance namespace, which has been defined for exactly this purpose; see [XS1], Section 2.6.1). Old clients should also ignore additional attributes on property elements returned by PROPFIND (and similar methods), although the WebDAV specification only defines this behaviour for unknown elements and is silent about unknown attributes (see [RFC2518], Section 23.3.2.2).

Servers not aware of datatype handling either drop the "xsi:type" attribute or have it persist along with the property value (see

[RFC2518], Section 4.4). However, they will never indicate successful parsing of the datatype by returning back the type in the response to PROPPATCH. Thus, clients can supply type information without having to poll for server support in advance.

8. Internationalization Considerations

This proposal builds on [RFC2518] and inherits its internationalizability.

9. Security Considerations

This protocol extension does not introduce any new security implications beyond those documented for the base protocol (see [RFC2518], Section 17).

10. Acknowledgements

This document has benefited from thoughtful discussion by Lisa Dusseault, Stefan Eissing, Eric Sedlar, and Kevin Wiggen.

11. References

11.1. Normative References

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

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., Carter, S., and D. Jensen, "HTTP Extensions for Distributed Authoring -- WEBDAV", RFC 2518, February 1999.

[XS1] Thompson, H., Beech, D., Maloney, M., Mendelsohn, N., and World Wide Web Consortium, "XML Schema Part 1: Structures Second Edition", W3C REC-xmlschema-1-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>.

[XS2] Biron, P., Malhotra, A., and World Wide Web Consortium, "XML Schema Part 2: Datatypes Second Edition", W3C REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>.

11.2. Informative References

[RFC3253] Clemm, G., Amsden, J., Ellison, T., Kaler, C., and J. Whitehead, "Versioning Extensions to WebDAV", RFC 3253, March 2002.

Author's Address

Julian F. Reschke
greenbytes GmbH
Hafenweg 16
Muenster, NW 48155
Germany

Phone: +49 251 2807760
Fax: +49 251 2807761
EMail: julian.reschke@greenbytes.de
URI: http://greenbytes.de/tech/webdav/

Full Copyright Statement

Copyright (C) The Internet Society (2005).

This document is subject to the rights, licenses and restrictions contained in BCP 78 and at www.rfc-editor.org/copyright.html, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.