Network Working Group                                          J. Reschke
Request for Comments: 4709                                      greenbytes
Category: Informational                                      October 2006


       Mounting Web Distributed Authoring and Versioning (WebDAV) Servers

Status of This Memo

Copyright Notice

Abstract

   In current Web browsers, there is no uniform way to specify that a
   user clicking on a link will be presented with an editable view of a
   Web Distinguished Authoring and Versioning (WebDAV) server.  For
   example, it is frequently desirable to be able to click on a link and
   have this link open a window that can handle drag-and-drop
   interaction with the resources of a WebDAV server.

   This document specifies a mechanism and a document format that
   enables WebDAV servers to send "mounting" information to a WebDAV
   client.  The mechanism is designed to work on any platform and with
   any combination of browser and WebDAV client, relying solely on the
   well-understood dispatch of documents through their MIME type.

Table of Contents

1.  Introduction

   By definition, a Web Distributed Authoring and Versioning (WebDAV)
   server ([RFC2518]) is an HTTP server as well ([RFC2616]).  Most
   WebDAV servers can be (at least partly) operated from an HTML-based
   user interface in a web browser.  However, it is frequently desirable
   to be able to switch from an HTML-based view to a presentation
   provided by a native WebDAV client, directly supporting the authoring
   features defined in WebDAV and related specifications.

   This document specifies a platform-neutral mechanism based on the
   dispatch of documents through their MIME type.  For completeness,
   Appendix A lists other approaches that have been implemented in
   existing clients.

   For example, many educational institutions use WebDAV servers as a
   mechanism for sharing documents among students.  Each student owns a
   separate collection structure on a WebDAV server, often called his/
   her "locker".  Ideally, when users click on a link in an HTML page
   provided by the university (perhaps by their university Web portal),
   an editable view of their locker will appear.

2.  Terminology

   The terminology used here follows that in the WebDAV Distributed
   Authoring Protocol specification [RFC2518].

   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].

   This document uses XML DTD fragments ([XML]) as a purely notational
   convention.  In particular:

   o  Element names use the namespace
      "http://purl.org/NET/webdav/mount".  When an XML element type in
      this namespace is referenced in this document outside of the
      context of an XML fragment, the string "dm:" will be prefixed to
      the element name.

   o  Element ordering is irrelevant.

   o  Extension elements/attributes (elements/attributes not already
      defined as valid child elements) may be added anywhere, except
      when explicitly stated otherwise.

3.  Format

   A WebDAV mount request is encoded in a specific XML format ([XML])
   with a well-defined MIME type (see Section 6.1).  The MIME type
   allows user agents to dispatch the content to a handler specific to
   the system's WebDAV client.

   The elements defined below use the namespace
   "http://purl.org/NET/webdav/mount".

   <!ELEMENT mount (url, open?, username?) >

   <!ELEMENT url (#PCDATA) >
   <!-- PCDATA value: scheme ":" hier-part, as defined in Section 3 of
        [RFC3986] -->

   <!ELEMENT open (#PCDATA) >
   <!-- PCDATA value: path, as defined in Section 3 of
        [RFC3986] -->

   <!ELEMENT username (#PCDATA) >

3.1.  dm:mount

   The <dm:mount> element acts as a container for all the remaining
   elements defined by this protocol.

3.2.  dm:url

   The mandatory <dm:url> element provides the HTTP URL of the WebDAV
   collection that should be mounted by the client.

3.3.  dm:open

   The optional <dm:open> element instructs the client to display the
   specified child collection; its URL is computed by concatenating this
   element's value with the URL obtained from the <dm:url> (Section 3.2)
   element (see Section 7 for a discussion about why this element only
   supports displaying collections rather than opening arbitrary
   documents).

3.4.  dm:username

   The server can use the optional <dm:username> element to specify the
   name of the currently authenticated principal.  A client can use this
   value to select a matching mount point (different users may have
   mounted the URL with different credentials under different local
   mount points) or to provide a meaningful default for authentication
   against the server.  It is common that a browser and WebDAV client do
   not share HTTP connections, so including this information in the
   mount document increases usability.

   Implementation Note: If a <dm:username> element is present, public
   caching of the document should be disallowed.  Thus, appropriate
   'Vary' or 'Cache-Control' headers are needed in the server response.

4.  Example

   In the example below, the client first retrieves a representation of
   a WebDAV collection using a generic Web browser (1).  The returned
   HTML content contains a hyperlink that identifies the "davmount"
   document in the format defined in Section 3 (2).  The user follows
   this link (3), which causes the server to return the "davmount"
   document to the user's browser (4).  The browser in turn passes the
   content to the application that was registered to handle the
   "application/davmount+xml" MIME type, usually the default WebDAV
   client on the client's system.

(1) Client retrieves representation of WebDAV collection "/user42/
inbox/".

```
GET /user42/inbox/ HTTP/1.1
Host: www.example.com
```

(2) Server returns representation.

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: xxx

..
<a href="?action=davmount">View this collection in your
WebDAV client</a>
..
```

(note that the example shows only that part of the HTML page that
contains the relevant link)

(3) Client follows link to "davmount" document

```
GET /user42/inbox/?action=davmount HTTP/1.1
Host: www.example.com
```

(4) Server returns "davmount" document

```
HTTP/1.1 200 OK
Content-Type: application/davmount+xml
Content-Length: xxx
Cache-Control: private

<dm:mount xmlns:dm="http://purl.org/NET/webdav/mount">
  <dm:url>http://www.example.com/user42/</dm:url>
  <dm:open>inbox/</dm:open>
</dm:mount>
```

5.  Internationalization Considerations

   This document does not introduce any new internationalization
   considerations beyond those discussed in [RFC2518], Section 16.

6.  IANA Considerations

6.1.  MIME Type Registration

   Type name:

      application

   Subtype name:

      davmount+xml

   Required parameters:

   Optional parameters:

      "charset": This parameter has identical semantics to the charset
      parameter of the "application/xml" media type as specified in
      [RFC3023].

   Encoding considerations:

      Identical to those of "application/xml" as described in [RFC3023],
      Section 3.2.

   Security considerations:

      As defined in this specification.  In addition, as this media type
      uses the "+xml" convention, it shares the same security
      considerations as described in [RFC3023], Section 10.

   Interoperability considerations:

      There are no known interoperability issues.

   Published specification:

      This specification.

   Applications that use this media type:

      SAP Netweaver Knowledge Management, Xythos Drive.

      Additional information:

         Magic number(s):

            As specified for "application/xml" in [RFC3023], Section 3.2.

         File extension(s):

            .davmount

         Fragment identifiers:

            As specified for "application/xml" in [RFC3023], Section 5.

         Base URI:

            As specified in [RFC3023], Section 6.

         Macintosh file type code(s):

            TEXT

      Person & email address to contact for further information:

         Julian Reschke <julian.reschke@greenbytes.de>

      Intended usage:

         COMMON

      Restrictions on usage:

         None.

      Author:

         Julian Reschke

      Change controller:

         IESG

7.  Security Considerations

   All security considerations connected to HTTP/WebDAV and XML apply
   for this specification as well, namely, [RFC2518] (Section 17) and
   [RFC3470] (Section 7).

   In addition, client implementers must be careful when implementing
   the <dm:open> element (see Section 3.3).  It MUST NOT be used to
   initiate any action beyond displaying the contents of a WebDAV
   collection (supporting "opening" documents could be abused to trick a
   user into letting the operating system's shell execute arbitrary
   content, possibly running it as an executable program).

   The OPTIONAL <dm:username> element defined in Section 3.4 allows the
   inclusion of user names into mount documents.  However in some cases,
   user name information is considered to be security sensitive.  Should
   this be the case, parties generating mount documents are advised to
   either not to include user names, or to use access control to
   restrict access to the information as desired.

8.  Acknowledgements

   This document has benefited from thoughtful discussion by Emile
   Baizel, Spencer Dawkins, Lisa Dusseault, Stefan Eissing, Joe
   Gregorio, Michal Gregr, Russ Housley, Jim Luther, Jaroslav Mazanec,
   and Jim Whitehead.

9.  References

9.1.  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.

   [RFC2616]   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.

   [RFC3023]   Murata, M., St.Laurent, S., and D. Kohn, "XML Media
               Types", RFC 3023, January 2001.

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

   [XML]       Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., and
               F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fourth
               Edition)", W3C REC-xml-20060816, August 2006,
               <http://www.w3.org/TR/2006/REC-xml-20060816>.

9.2.  Informative References

   [RFC3470]   Hollenbeck, S., Rose, M., and L. Masinter, "Guidelines for
               the Use of Extensible Markup Language (XML) within IETF
               Protocols", RFC 3470, BCP 70, January 2003.

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

Appendix A.  Alternative Approaches

A.1.  ...Through HTML/CSS Extensions

   Microsoft Internet Explorer implements a Cascading Style Sheet (CSS)
   extension that allows switching to its own WebDAV client
   ("Webfolder", see <http://msdn.microsoft.com/workshop/author/
   behaviors/reference/behaviors/anchor.asp>).  However, at the time of
   this writing, this extension was not implemented by any other user
   agent.

A.2.  ...Through Custom URI Schemes

   The "kio" library of the "K Desktop Enviroment"
   (<http://www.kde.org/>) uses the URI scheme "webdav" to dispatch to
   the system's WebDAV client.  This URI scheme is not registered, nor
   is it supported on other platforms.  Furthermore, the W3C's
   "Architecture of the World Wide Web, Volume One" explicitly advises
   against defining new schemes when existing schemes can be used:

      A specification SHOULD reuse an existing URI scheme (rather than
      create a new one) when it provides the desired properties of
      identifiers and their relation to resources.

   (See [WEBARCH], Section 2.4.)

Appendix B.  Implementations

B.1.  Example Implementation for Webfolder Client

   The figure below shows a sample implementation of a dispatcher for
   the application/davmount+xml datatype, suited for Win32 systems and
   the Microsoft "Webfolder" client.

   ```
   // sample implementation of application/davmount+xml
   // dispatcher for Windows Webfolder client
   //
   // to install/uninstall:
   //        wscript davmount.js
   //
   // to open the webfolder:
   //        wscript davmount.js filename
   // (where filename refers to an XML document with MIME type
   // application/davmount+xml)

   var EXTENSION = ".davmount";
   var MIMETYPE = "application/davmount+xml";
   var REGKW = "WebDAV.mount";
   ```

```
   var NS = "xmlns:m='http://purl.org/NET/webdav/mount";

   // remove keys/entries from the registry

   function regdel(shell, key) {
     try {
       var x = shell.RegRead(key);
       try {
         shell.RegDelete(key);
       }
       catch(e) {
         WScript.Echo("Error removing key " + key + ": " + e);
       }
     }
     catch(e) {
       // entry not present
     }
   }


   // methods for registering/unregistering the handler

   function install() {

     var WshShell = new ActiveXObject("WScript.Shell");
     if (WshShell == null) {
       WScript.Echo("Couldn't instantiate WScript.Shell object");
       return 2;
     }

     var fso = new ActiveXObject("Scripting.FileSystemObject");

     var RegExt = "HKCR\\" + EXTENSION + "\\";
     var RegMimeType = "HKCR\\MIME\\DataBase\\Content Type\\"
       + MIMETYPE + "\\";
     var RegKw = "HKCR\\" + REGKW + "\\";

     var extension = null;
     try {
       extension = WshShell.RegRead(RegMimeType + "Extension");
     }
     catch (e) {
     }

     if (extension == null) {
       var but = WshShell.popup("Install the dispatcher for mime type "
         + MIMETYPE + "?", 0, MIMETYPE + " installation", 4);
```

```
      if (but == 6) {
        try {
          WshShell.RegWrite(RegExt, REGKW);
          WshShell.RegWrite(RegExt + "Content Type", MIMETYPE);
          WshShell.RegWrite(RegMimeType + "Extension", EXTENSION);
          WshShell.RegWrite(RegKw, "WebDAV Mount Request");
          WshShell.RegWrite(RegKw + "DefaultIcon\\",
            "shell32.dll,103");
          var path = fso.getAbsolutePathName("davmount.js");
          WshShell.RegWrite(RegKw + "shell\\open\\command\\",
            "%SystemRoot%\\system32\\wscript.exe /nologo \""
            + path + "\" \"%1\"", "REG_EXPAND_SZ");
        }
        catch (e) {
          WScript.Echo("Error writing to registry");
          return 1;
        }

        return 0;
      }
      else {
        return 1;
      }
    }
    else {
      var but = WshShell.popup("Remove the dispatcher for mime type "
        + MIMETYPE + "?", 0, MIMETYPE + " installation", 4);

      if (but == 6) {
        regdel(WshShell, RegExt + "Content Type");
        regdel(WshShell, RegExt);
        regdel(WshShell, RegKw + "shell\\open\\command\\");
        regdel(WshShell, RegKw + "DefaultIcon\\");
        regdel(WshShell, RegKw);
        regdel(WshShell, RegMimeType + "Extension");
        regdel(WshShell, RegMimeType);
        return 0;
      }
      else {
        return 1;
      }
    }
  }


  if (WScript.Arguments.length == 0) {
    // install/uninstall
    WScript.Quit(install());
```

```
      }
    else {
      // try to invoke Webfolder

      var inp = new ActiveXObject("MSXML2.DOMDocument");
      var furi = encodeURI(WScript.Arguments(0));
      if (! inp.load(furi)) {
        WScript.Echo("Can't read from '"
          + WScript.Arguments(0) + "'!");
        WScript.Quit(2);
      }

      inp.setProperty("SelectionLanguage", "XPath");
      inp.setProperty("SelectionNamespaces",
        "xmlns:m='http://purl.org/NET/webdav/mount'");

      var n1 = inp.selectSingleNode("/m:mount/m:url");
      var n2 = inp.selectSingleNode("/m:mount/m:open");

      if (n1 == null) {
        WScript.Echo("<url> element missing.");
        WScript.Quit(2);
      }

      var ie = new ActiveXObject("InternetExplorer.Application");

      ie.Navigate("about:blank");
      var doc = ie.Document;

      var folder = doc.createElement("span");
      folder.addBehavior("#default#httpFolder");

      var result = folder.navigate(n1.text +
                    (n2 == null ? "" : n2.text));

      // close the window again when there was no <open> element
      if (n2 == null) ie.Quit();

      if (result != "OK") {
        if (result == "PROTOCOL_NOT_SUPPORTED") {
          WScript.Echo("This site doesn't seem to support WebDAV.");
          WScript.Quit(1);
        }
        else {
          WScript.Echo("Unexpected status: " + result);
          WScript.Quit(2);
        }
      }
```

```
   }
```

B.2.  Xythos

   The "Xythos Drive" WebDAV client for WebDAV supports this
   specification starting with version 4.4.

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/