[MC-FPSEWM]:

FrontPage Server Extensions: Website Management Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
8/10/2007	0.1	Major	Initial Availability
9/28/2007	0.2	Minor	Updated the technical content.
10/23/2007	0.3	Minor	Updated the technical content.
11/30/2007	0.4	Minor	Updated links and performed glossary term tagging.
1/25/2008	0.4.1	Editorial	Revised and edited the technical content.
3/14/2008	1.0	Major	Updated and revised the technical content.
5/16/2008	1.0.1	Editorial	Revised and edited the technical content.
6/20/2008	1.1	Minor	Updated the technical content.
7/25/2008	1.1.1	Editorial	Revised and edited the technical content.
8/29/2008	2.0	Major	Updated and revised the technical content.
10/24/2008	2.1	Minor	Updated the technical content.
12/5/2008	3.0	Major	Updated and revised the technical content.
1/16/2009	3.0.1	Editorial	Revised and edited the technical content.
2/27/2009	3.0.2	Editorial	Revised and edited the technical content.
4/10/2009	3.0.3	Editorial	Revised and edited the technical content.
5/22/2009	3.1	Minor	Updated the technical content.
7/2/2009	4.0	Major	Updated and revised the technical content.
8/14/2009	4.1	Minor	Updated the technical content.
9/25/2009	4.2	Minor	Updated the technical content.
11/6/2009	4.2.1	Editorial	Revised and edited the technical content.
12/18/2009	4.2.2	Editorial	Revised and edited the technical content.
1/29/2010	4.3	Minor	Updated the technical content.
3/12/2010	4.3.1	Editorial	Revised and edited the technical content.
4/23/2010	5.0	Major	Updated and revised the technical content.
6/4/2010	6.0	Major	Updated and revised the technical content.
7/16/2010	6.1	Minor	Clarified the meaning of the technical content.
8/27/2010	6.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2010	7.0	Major	Significantly changed the technical content.
11/19/2010	8.0	Major	Significantly changed the technical content.
1/7/2011	8.0	None	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
2/11/2011	8.0	None	No changes to the meaning, language, or formatting of the technical content.
3/25/2011	8.0	None	No changes to the meaning, language, or formatting of the technical content.
5/6/2011	8.0	None	No changes to the meaning, language, or formatting of the technical content.
6/17/2011	8.1	Minor	Clarified the meaning of the technical content.
9/23/2011	8.1	None	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	8.1	None	No changes to the meaning, language, or formatting of the technical content.
3/30/2012	8.1	None	No changes to the meaning, language, or formatting of the technical content.
7/12/2012	8.1	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	8.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	8.1	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2013	8.1	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	8.1	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	8.1	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	8.1	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	8.1	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	8.1	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	8.1	None	No changes to the meaning, language, or formatting of the technical content.
6/23/2016	8.1	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	8.1	None	No changes to the meaning, language, or formatting of the technical content.
7/24/2018	8.2	Minor	Clarified the meaning of the technical content.
10/1/2018	8.3	Minor	Clarified the meaning of the technical content.

Date	Revision History	Revision Class	Comments
12/11/2018	8.4	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction

The FrontPage Server Extensions: Website Management Protocol specifies a set of server extensions that can be used to augment a basic **Hypertext Transfer Protocol (HTTP)** server. These extensions provide file server functionality similar to **Web Distributed Authoring and Versioning Protocol (WebDAV)**, allowing a **website** to be presented as a file share. The use of WebDAV is recommended over the FrontPage Server Extensions: Website Management Protocol. For more information about WebDAV, see [MS-WDVME].

The FrontPage Server Extensions: Website Management Protocol uses HTTP version 1.1 (as described in [RFC2616]) as a transport. Requests are specialized HTTP POSTs or GETs, and responses are in HTML, as described in [RFC2854]. Despite the use of HTTP, the protocol is intended to be used by a client program, not by the user directly through a web browser.

The FrontPage Server Extensions: Website Management Protocol is a superset of a smaller protocol known as FrontPage Server Extensions Remote Protocol, as described in [MS-FPSE]. The FrontPage Server Remote Protocol Extensions is the protocol that is used when communicating between Microsoft Windows® clients and Windows servers. The larger protocol is used to perform a wider array of website administration tasks, including theme management, site navigation, and document repository tasks.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

bot: A structured HTML comment that is processed by a front-end web server when the containing document is opened by or saved to the server. Also referred to as web bot.

cascading style sheet (CSS): An extension to HTML that enables authors and users of HTML documents to attach style sheets to those documents, as described in [CSS-LEVEL1] and [CSS-LEVEL2]. A style sheet includes typographical information about the appearance of a page, including the font for text on the page.

dictionary: A collection of key/value pairs. Each pair consists of a unique key and an associated value. Values in the dictionary are retrieved by providing a key for which the dictionary returns the associated value.

document: An object in a content database such as a file, folder, **list**, or site. Each object is identified by a URI.

document library: A type of list that is a container for documents and folders.

domain: A set of users and computers sharing a common namespace and management infrastructure. At least one computer member of the set must act as a domain controller (DC) and host a member list that identifies all members of the domain, as well as optionally hosting the Active Directory service. The domain controller provides authentication of members, creating a unit of trust for its members. Each domain has an identifier that is shared among its members. For more information, see [MS-AUTHSOD] section 1.1.1.5 and [MS-ADTS].

domain name: The name given by an administrator to a collection of networked computers that share a common directory. Part of the domain naming service naming structure, domain names consist of a sequence of name labels separated by periods.

field internal name: A string that uniquely identifies a field in a content type or a SharePoint list.

folder: A file system construct. File systems organize a volume's data by providing a hierarchy of objects, which are referred to as folders or directories, that contain files and can also contain other folders.

form: A structured document with controls and spaces that are reserved for entering and displaying information. Forms can contain special coding for actions such as submitting and querying data.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

language code identifier (LCID): A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.

link fixup: A process that helps to ensure consistent paths to linked components.

list: A container within a SharePoint site that stores list items. A list has a customizable schema that is composed of one or more fields.

list template: An XML-based definition of list settings, including fields and views, and optionally list items. List templates are stored in .stp files in the content database.

manifest: A file that stores metadata about an expansion pack, such as the name of the expansion pack, the files and resources that are included in the expansion pack, and the dependencies that it has on other files and components.

master page: An ASP.NET file that has a predefined layout that can include static text, HTML elements, and server controls.

metadict: A **dictionary** that has strongly typed values.

page: A file that consists of HTML and can include references to graphics, scripts, or dynamic content such as Web Parts.

server-relative URL: A relative URL that does not specify a scheme or host, and assumes a base URI of the root of the host, as described in [RFC3986].

service: A process or agent that is available on the network, offering resources or services for clients. Examples of services include file servers, web servers, and so on.

site: A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as web site.

site collection: A set of **websites** that are in the same content database, have the same owner, and share administration settings. A site collection can be identified by a **GUID** or the URL of

the top-level site for the site collection. Each site collection contains a top-level site, can contain one or more subsites, and can have a shared navigational structure.

subsite: A complete website that is stored in a named subdirectory of another website. The parent website can be the top-level site of a site collection or another subsite. Also referred to as subweb.

thicket: A means of storing a complex HTML document with its related files. It consists of a thicket main file and a hidden thicket folder that contains a thicket manifest and a set of thicket supporting files that, together, store the referenced content of the document.

Unicode: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

UTF-8: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

web bot: See **bot**.

Web Distributed Authoring and Versioning Protocol (WebDAV): The Web Distributed Authoring and Versioning Protocol, as described in [RFC2518] or [RFC4918].

Web Part: A reusable component that contains or generates web-based content such as XML, HTML, and scripting code. It has a standard property schema and displays that content in a cohesive unit on a webpage. See also Web Parts Page.

website: A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as site.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-WDVSE] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Server Extensions".

[MSDN-ThemeDef] Microsoft Corporation, "Theme Definition", http://msdn.microsoft.com/en-us/library/ms965733.aspx

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", RFC 1123, October 1989, http://www.ietf.org/rfc/rfc1123.txt

[RFC1341] Borenstein, N., and Freed, N., "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, June 1992, http://www.rfc-editor.org/rfc/rfc1341.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2279] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998, http://www.rfc-editor.org/rfc/rfc2279.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC2854] Connolly, D. and Masinter, L., "The 'text/html' Media Type", RFC 2854, June 2000, http://www.ietf.org/rfc/rfc2854.txt

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, http://www.rfc-editor.org/rfc/rfc3986.txt

[RFC4234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005, http://www.rfc-editor.org/rfc/rfc4234.txt

1.2.2 Informative References

[MS-FPSE] Microsoft Corporation, "FrontPage Server Extensions Remote Protocol".

[MS-WDVME] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Microsoft Extensions".

1.3 Overview

The FrontPage Server Extensions: Website Management Protocol is used by client applications to display and modify the contents of a **site**. The FrontPage Server Extensions: Website Management Protocol uses a set of methods on a web server to provide file uploading and downloading, directory creation and listing, basic file locking, and file movement.

Each message from the client is in the format of an HTTP POST or GET as described in [RFC2616] sections 9.5 and 9.3, that includes a set of parameters, and each reply from the server returns a set of values as an HTML response, as described in [RFC2854]. The method parameter defines what operation the server will perform in addition to the meanings of the other parameters and return values.

The client sends method call requests to the server, and the server sends return values to the client via HTML. The server never initiates any communication with the client. All communication is transported over HTTP or secure HTTP (HTTPS), as described in [RFC2616] section 9.1. Method calls are sent as HTTP POSTs with the method name and arguments as message headers (described in [RFC2616] section 4.2), and server responses are sent as a list in the message body (described in [RFC2616] section 4.3) of an HTTP response. All posts are made to one of several well-defined URLs on the server, which can be discovered by clients.

The following sequence diagram depicts a generic FrontPage Server Extensions conversation. A brief explanation of each message follows, and details are defined in sections 2 and 3.

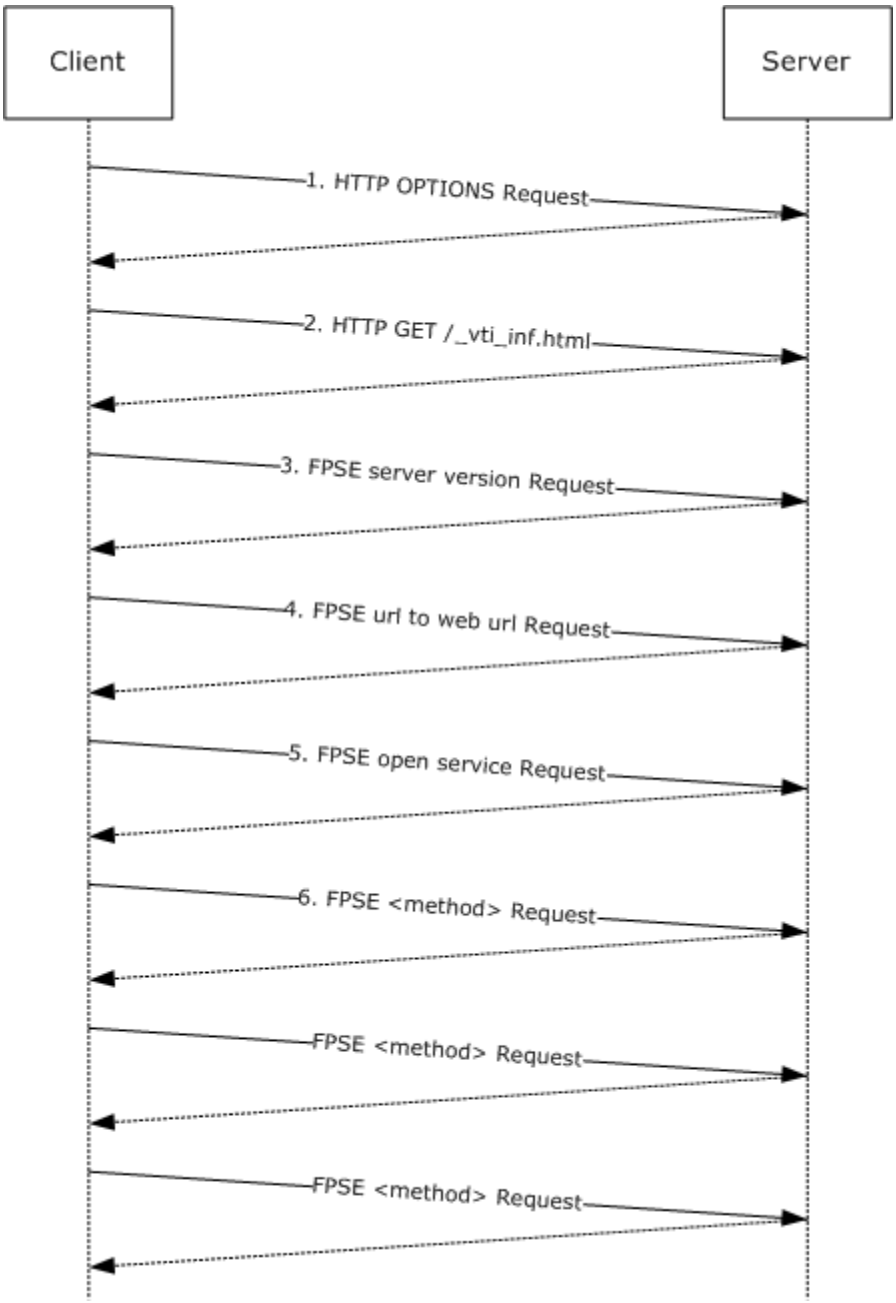

Figure 1: Generic FrontPage Server Extension message sequence

1. The HTTP OPTIONS request is sent to determine if the server supports the FrontPage Server Extensions: Website Management Protocol. If the response contains the MS-Author-Via header (as described in section 3.1.3.1), the server supports the protocol. Clients often cache this value.

2. The HTTP GET on _vti_inf.html returns information that specifies the well-defined URLs to which the client POST further method calls.

3. At this point, the client is prepared to start making method calls against the server. The first call is a server version (section 3.1.5.3.36) request whereby the client negotiates a protocol version with the server.

4. The client can then call url to web url (section 3.1.5.3.41) if the site is a **subsite**, that is, not located at the root of the server's namespace.

5. Then, the client can make an open service (section 3.1.5.3.24) request on the site that it wants to open. This request is optional, but it will return information about the site's capabilities, such as support for version control.

6. The client can make any method calls against the server. The nature of any further client-server communication is determined by the specific actions of the client at the time.

1.4 Relationship to Other Protocols

The FrontPage Server Extensions: Website Management Protocol is transported via HTTP version 1.1 GETs, POSTs and responses, as described in [RFC2616] sections 9.3, 9.5, and 6, respectively.

1.5 Prerequisites/Preconditions

The client knows the URL of the server that it wants to communicate with, which is usually passed by the user as the prompt for beginning the FrontPage Server Extensions: Website Management Protocol conversation. If required by the server, the client authenticates by using the underlying HTTP mechanisms, as described in [RFC2616] section 14.8.

1.6 Applicability Statement

The FrontPage Server Extensions: Website Management Protocol is a precursor to the WebDAV protocol and can be used in similar situations. Because the FrontPage Server Extensions: Website Management Protocol is an earlier technology, most implementers will find WebDAV, as described in [MS-WDVME], a more appealing option.

1.7 Versioning and Capability Negotiation

1.7.1 Protocol Versions

Version negotiation is performed by using the server version (section 3.1.5.3.36) method. The client sends its own protocol version in the method name section of the request. The server compares that to the server protocol version and replies to the client. The protocol version that the server uses is given in the response header in the format of (Min(ServerVersion, ClientVersion)). The client is expected to use this version for any remaining communications. If the version of the client or server is not supported, the one with the newer protocol version discontinues the conversation.<1>

The structure of a FrontPage Server Extensions: Website Management Protocol version (as described in section 2.2.2.2.9), as defined, has four parts: a major version, a minor version, a phase number, and a build number. Therefore, a version might look like 1.0.0.3214. Versions grow over time, so 3.0 is considered earlier, or older, than 4.0.

In the FrontPage Server Extensions: Website Management Protocol, the client, server, and protocol each have their own version, although all of them follow the same format. The client and server version are used in the negotiation to determine the protocol version. For details, see section 3.1.5.3.36.

All servers reject any client with a version earlier than 4.0.2.2611, and clients reject any server with a version earlier than 3.0.2.1002. The server returns a V_RPC_CLIENT_TOO_OLD (0x0004000C) error code (see section 2.2.2.2.17.1) if an incompatible client is encountered. If the version of the server is not supported, the client simply ignores the server, and no further communication with the server is attempted.

1.7.2 Capability Negotiation

The Microsoft FrontPage Server Extensions clients and servers perform capability negotiation because some operations are supported only by newer servers. This negotiation is performed by using the site metadata that is returned in the server version (section 3.1.5.3.36) method. Clients can determine server capabilities by looking for certain values in the metadata that specify the version-specific behaviors that the server supports. The capability metadata values are stored with metakeys detailed in section 2.

1.8 Vendor-Extensible Fields

There are no vendor-extensible fields in the FrontPage Server Extensions: Website Management Protocol.

1.9 Standards Assignments

The FrontPage Server Extensions: Website Management Protocol does not use any standards assignments other than those of HTTP 1.1, as described in [RFC2616].

2 Messages

The following sections specify FrontPage Server Extensions: Website Management Protocol message transport and message syntax. This protocol references commonly used data types as defined in [MS-DTYP].

2.1 Transport

The FrontPage Server Extensions: Website Management Protocol uses HTTP version 1.1 (as specified in [RFC2616]) as transport for the GET and POST methods.

2.1.1 Client Requests

Client requests to the server MUST be transmitted as POST or GET methods appended to a URL, hereafter referred to as the URL Mode. For details about the syntax, see section 2.2.1.

If the client request does not conform to the message definitions that follow, the server MUST return a syntax error to the client and stop parsing the request. See section 3.1.5.2 for details on the format of server error responses.

2.1.2 Server Responses

Server responses to client requests MUST be transmitted as HTML (as specified in [RFC2854]) and are hereafter referred to as HTML Mode. Exceptions are if the server responses are otherwise specified. For details about the syntax, see section 2.2.1.

If the server response does not conform to the message definitions that follow, the client MUST ignore the server response and stop communication with the server.

2.2 Message Syntax

This section specifies the syntax and the data types that are used when a Microsoft Windows® client posts FrontPage Server Extensions: Website Management Protocol requests to a server. It also specifies the syntax that is used by the server to respond to client requests. The syntax and data types are defined using **ABNF**, as specified in [RFC4234].

2.2.1 Syntax

The FrontPage Server Extensions: Website Management Protocol is used in URL Mode and HTML Mode in client requests and server responses, respectively. These two modes differ with respect to encoding rules and the values of certain tokens in the stream. Implementations MUST use the following syntax rules that define these encoding schemes.

All FrontPage Server Extensions: Website Management Protocol communications are case-sensitive. The reader SHOULD assume that all strings are case-sensitive unless otherwise noted.

2.2.1.1 Syntax Delimiters

The following two sections specify primitives that are used as punctuation within strings in the full syntax for both URL Mode and HTML Mode, respectively. They are defined for both URL Mode and HTML Mode, so that in the remainder of this protocol document a single definition can be given for higher-level constructs.

2.2.1.1.1 URL Mode

The following delimiters are used as punctuation within a string in URL Mode.

```
PARGSEP = "&"
SARGSEP = ";"
VALSEP = "="
LISTSEP = ";"
OBRACKET = "["
CBRACKET = "]"
STARTLIST = ""
```

2.2.1.1.2 HTML Mode

The following delimiters are used as punctuation within a string in HTML Mode.

```
PARGSEP = LF "<p>"
SARGSEP = LF "<li>"
VALSEP = "="
LISTSEP = LF "<li>"
OBRACKET = LF "<ul>"
CBRACKET = LF "</ul>"
STARTLIST = LF "<li>"
```

2.2.1.1.3 Nesting Level Dependent Elements

An implementation of the FrontPage Server Extensions: Website Management Protocol MUST keep track of the number of times an OBRACKET is sent minus the number of times a CBRACKET is sent in the current request. Hereafter, this value is referred to as the nesting level. This value affects which delimiters are used.

If the nesting level 0, the following delimiters are used.

```
ARGSEP = PARGSEP
```

Otherwise, if the nesting level is not 0, the following delimiters are used.

```
ARGSEP = SARGSEP
```

2.2.1.2 Character Escaping

The FrontPage Server Extensions: Website Management Protocol uses Unicode Translation Format-8 (**UTF-8**) (as specified in [RFC2279]) as its character encoding. In every instance that follows in this protocol document in which a string is referred to as a literal, it can be assumed that the character is UTF-8 encoded. Depending on the mode, URL or HTML, various character escaping is used, as shown in the following sections.

2.2.1.2.1 URL Mode

In URL Mode, characters are escaped as follows.

```
ESCAPED-BYTE = ALPHA / DIGIT ; used with the literal meaning
  / "+" ; encoded space
  / "%5c%5c" ; encoded backslash
  / "%5c%3d" ; encoded equal sign
  / "%5c%5b" ; encoded open bracket
  / "%5c%5d" ; encoded close bracket
```

```
  / "%5c%3b" ; encoded semicolon
  / "%5c%22" ; encoded double quote
  / "%" 2HEXDIG ; used to encode anything not mentioned above
```

A sender SHOULD encode in this order (for example, a space SHOULD be encoded as "+" rather than "%20"; a capital "A" SHOULD be encoded as "A" rather than "%41"). A receiver MUST decode both "%20" and "+" to a space. A backslash MUST be ignored except in the following two cases:

When it is followed by another backslash, the pair MUST be treated as a single literal backslash.

When it is followed by an equal sign (=), an opening bracket ([), a closing bracket (]), or a semicolon (;), the backslash MUST be ignored, but the character that comes after the backslash MUST NOT be treated as a delimiter.

2.2.1.2.2 HTML Mode

In HTML Mode, characters are escaped as follows.

```
ESCAPED-BYTE =
  %d32-33 / %d35-58 / %d63-91 ; literal meaning
  / %d93-122 / %d124 / %d126-127 ; literal meaning
  / "\t" ; encoded tab (%d8)
  / "\b" ; encoded backspace (%d9)
  / "\n" ; encoded newline (%d10)
  / "\f" ; encoded formfeed (%d12)
  / "\r" ; encoded carriage return (%d13)
  / "&#" 2DIGIT ";" ; encoded non-printing characters that are not
                      specially handled (%d0-7 / %d11 / %d14-31) or
                      special printing characters (%d34 / %d59-62 / %d92)
  / "&#" 3DIGIT ";" ; special printing characters (%d123 / %d125) or
                      non-printing 3 digit characters (%d128-255)
```

For example, send "\t" (third expansion) rather than "&#08;" (eighth expansion), and send "<" (eighth expansion) rather than "<" (ninth expansion). However, a receiver MUST accept any of these forms.

2.2.2 Data Types

This section describes the data types that are used when the Microsoft Windows® client posts FrontPage Server Extensions: Website Management Protocol requests to the server, and the server responds to the client.

2.2.2.1 Primitive Data Types

This section specifies the primitive data types that are used in the FrontPage Server Extensions: Website Management Protocol, using **ABNF** as specified in [RFC4234].

2.2.2.1.1 UNSIGNED-INT

The UNSIGNED-INT data type is an unsigned integer which can be represented in 32 bits.

```
UNSIGNED-INT = 1*DIGIT ; default value = "0"
```

2.2.2.1.2 INT

The INT data type is a signed integer which can be represented in 32 bits.

```
INT = [ "-" ]  UNSIGNED-INT
```

2.2.2.1.3 BOOLEAN

The BOOLEAN data type represents a value which can be true or false.

```
TRUE = "true"
FALSE = "false"
BOOLEAN = TRUE / FALSE ; default value = FALSE
```

2.2.2.1.4 DOUBLE

The DOUBLE data type is a signed floating-point number which can be represented in 64 bits.

```
DOUBLE = INT [ "." UNSIGNED-INT ] / [ "-" ] "." UNSIGNED-INT
```

2.2.2.1.5 STRING

The STRING data type is an encoded text string of arbitrary length.

```
STRING = *ESCAPED-BYTE
```

A STRING represents a Unicode string with each ESCAPED-BYTE corresponding to a byte in a UTF-8 sequence. For example, the "æ" character (a combined "ae") is "U+00e6", which has a UTF-8 representation of "%xc3.a6". Therefore, the string "Cæsar" can be represented as "C%c3%a6sar" in URL Mode and as "CÃ¦sar" in HTML Mode.

2.2.2.1.6 TIME

The TIME data type is a string containing a date and time.

```
TIME = STRING
```

TIME values MUST conform to the Greenwich Mean Time (GMT) format, as specified in [RFC1123] section 5.2.14.

2.2.2.1.7 FILESYSTEMTIME

The FILESYSTEMTIME data type is a string containing an encoded **FILETIME** (specified in [MS-DTYP] section 2.3.3), split into a high-order 32-bit part and a low-order 32-bit part for serialization.

```
HIGHTIMEPART = "0x" 8HEXDIG
LOWTIMEPART = "0x" 8HEXDIG
FILESYSTEMTIME = HIGHTIMEPART "|" LOWTIMEPART
```

2.2.2.2 Complex Data Types

This section specifies the complex data types that are used in method requests and responses. These values, in addition to the primitive data types, are used throughout section 3.1.5.3 to define the data types for arguments and return values.

2.2.2.2.1 Vector

A **VECTOR** is a typed array of elements whose default value is empty.

```
VECTOR-UNSIGNED-INT = OBRACKET STARTLIST 1*(UNSIGNED-INT LISTSEP) CBRACKET
VECTOR-INT = OBRACKET STARTLIST 1*(INT LISTSEP) CBRACKET
VECTOR-BOOLEAN = OBRACKET STARTLIST 1*(BOOLEAN LISTSEP) CBRACKET
VECTOR-DOUBLE = OBRACKET STARTLIST 1*(DOUBLE LISTSEP) CBRACKET
VECTOR-STRING = OBRACKET STARTLIST 1*(STRING LISTSEP) CBRACKET
VECTOR-TIME = OBRACKET STARTLIST 1*(TIME LISTSEP) CBRACKET
VECTOR-FILESYSTEMTIME = OBRACKET STARTLIST 1*(FILESYSTEMTIME LISTSEP) CBRACKET


VECTOR-X = OBRACKET STARTLIST 1*(X LISTSEP) CBRACKET
```

All data types can have a vector type associated with them where X, as in the example in this section, represents the vector data type; for example, VECTOR-DOCINFO = OBRACKET STARTLIST 1*(DOCINFO LISTSEP) CBRACKET. X can be a simple type, such as **STRING** (section 2.2.2.1.5), or a complex type, such as **DOCINFO** (section 2.2.2.2.12).

2.2.2.2.2 Protocol-Version-String

A **PROTOCOL-VERSION-STRING** is an identifier for a specific protocol version, used for version negotiation between clients and servers.

```
PROTOCOL-VERSION-STRING = UNSIGNED-INT "." UNSIGNED-INT "."
   UNSIGNED-INT "." UNSIGNED-INT
```

2.2.2.2.3 URL-String

A **URL-STRING** is a URL in the format of a URI-reference, as specified in [RFC3986].

```
URL-STRING = URI-reference
VECTOR-URL-STRING = OBRACKET STARTLIST 1*(URL-STRING LISTSEP) CBRACKET
```

The **URL-STRING** can be further qualified as server-relative or service-relative for specific uses.

2.2.2.2.4 Request-Name-String

A **REQUEST-NAME-STRING** is a specifier for a method.

```
REQUEST-NAME-STRING = STRING
```

The **REQUEST-NAME-STRING** MUST be an encoded string containing one of the method name values defined in section 3.1.5.3.

2.2.2.2.5 RPCKEY and RPCVALUE

An **RPCKEY and RPCVALUE** pair are used to specify methods, parameters, and results.

```
RPCKEY-KEY-STRING = STRING
RPCKEY = [ARGSEP] RPCKEY-KEY-STRING VALSEP
```

The leading ARGSEP MUST be present in an RPCKEY, except in URL Mode when it is the first key after an OBRACKET or at the start of a response, in which case it MUST NOT be present. <2>

```
RPCVALUE = UNSIGNED-INT / INT / BOOLEAN / DOUBLE / STRING
   / TIME / FILESYSTEMTIME / VERSION / URL-STRING
   / METHOD-VALUE / DICT / METADICT / DOCINFO
   / DOCUMENT-LIST-RETURN-TYPE / SERVICE-RETURN-TYPE
   / DOC-INFO-REQUEST / URL-DIRECTORY / STATUS
   / PUT-OPTION / RENAME-OPTION
   / VECTOR-UNSIGNED-INT / VECTOR-INT
   / VECTOR-BOOLEAN / VECTOR-DOUBLE / VECTOR-STRING
   / VECTOR-URL-STRING / VECTOR-URL-DIRECTORY
   / section 2.2.2.2.22 / VECTOR-METADICT
   / VECTOR-ELEMENT-ID / VECTOR-STRUCTURE-ELEMENT
   / VECTOR-X
```

2.2.2.2.6 Method-Key-Value

The **METHOD-KEY-VALUE** is an **RPCKEY and RPCVALUE** pair (section 2.2.2.2.5) which specifies the method used by the server.

```
METHOD-KEY = RPCKEY
METHOD-VALUE = REQUEST-NAME-STRING [":" PROTOCOL-VERSION-STRING]
METHOD-KEY-VALUE = METHOD-KEY METHOD-VALUE
```

The RPC-KEY-STRING (section 2.2.2.2.5) in the RPCKEY of a METHOD-KEY MUST be "method".

2.2.2.2.7 Request Syntax

This section specifies the syntax for a FrontPage Server Extensions: Website Management Protocol request. A REQUEST consists of a method specifier which can be followed by parameter names with arguments. For details about which arguments ought to be sent for each method, refer to section 3.1.5.3.1.

```
REQUEST = METHOD-KEY-VALUE *(ARG-NAME ARG-VALUE) LF
```

The parameter names and arguments for the request are the set of ARG-NAME ARG-VALUE elements that appear after the METHOD-KEY-VALUE (section 2.2.2.2.6).

```
ARG-NAME = RPCKEY
ARG-VALUE = RPCVALUE
```

The **METHOD-KEY-VALUE** elements, and the **RPCKEY and RPCVALUE** pair are as specified in section 2.2.2.2.5.

2.2.2.2.8 Response Syntax

This section specifies the syntax for a FrontPage Server Extensions: Website Management Protocol response. The specific return values that ought to be sent for each method are specified in section 3.1.5.3.

```
RESPONSE = "<html><head><title>Vermeer RPC packet</title><head>" LF
  "<body>" METHOD-KEY-VALUE *(RET-NAME RET-VALUE)"</body>" LF "</html>"
  LF
```

The return values are the set of RET-NAME RET-VALUE elements that appear after the **METHOD-KEY-VALUE** (section 2.2.2.2.6).

```
RET-NAME = RPCKEY
RET-VALUE = RPCVALUE
```

The **METHOD-KEY-VALUE** elements, **RPCKEY and RPCVALUE** are as specified in section 2.2.2.2.5.

2.2.2.2.9 Version

Used to communicate a version number. The default value is "0.0.0.0".

```
VERSION = OBRACKET "major ver" VALSEP INT ARGSEP "minor ver" VALSEP
   INT ARGSEP "phase ver" VALSEP INT ARGSEP "ver incr" VALSEP INT
   CBRACKET
```

In phase version, the values are 0, 1, 2, or 3. The number 0 represents an alpha release or earlier; 1 represents a beta release; 2 represents an official release; and 3 represents a patched version increment that is used to differentiate, for example, SP1 from SP2, or internal builds before release.

Version numbers are ordered numerically, not lexicographically. For example, 12.9 is earlier than 12.10.

2.2.2.2.10 DICT

The FrontPage Server Extensions: Website Management Protocol format of a **dictionary** is a **DICT**.

```
KEY-STRING = STRING ; The key that is used to look up the value.
VALUE-STRING = STRING ; The data value that is looked up with the key.
DICT = OBRACKET [STARTLIST KEY-STRING LISTSEP VALUE-STRING *(LISTSEP
   KEY-STRING LISTSEP VALUE-STRING)] CBRACKET ; default value = empty
```

2.2.2.2.11 METADICT

The FrontPage Server Extensions: Website Management Protocol format of a metadictionary is a **METADICT**.

```
METADICT = DICT ; default value = empty
VECTOR-METADICT = OBRACKET STARTLIST 1*(METADICT LISTSEP) CBRACKET
```

For **METADICT**s, the VALUE-STRING (as specified in **DICT** (section 2.2.2.2.10), when decoded, MUST be in a format represented as METADICT-VALUE:

```
METADICT-VALUE = "T" METADICT-CONSTRAINT-CHAR "|" TIME
   / "V" METADICT-CONSTRAINT-CHAR "|" METADICT-STRING-VECTOR
   / "B" METADICT-CONSTRAINT-CHAR "|" BOOLEAN
   / "D" METADICT-CONSTRAINT-CHAR "|" DOUBLE
   / "I" METADICT-CONSTRAINT-CHAR "|" INT
   / "S" METADICT-CONSTRAINT-CHAR "|" STRING
   / "L" METADICT-CONSTRAINT-CHAR "|" STRING  ; more than 255 Unicode chars
   / "F" METADICT-CONSTRAINT-CHAR "|" FILESYSTEMTIME
   / "U" METADICT-CONSTRAINT-CHAR "|" METADICT-INT-VECTOR

METADICT-CONSTRAINT-CHAR = "X" / "R" / "W"
```

The METADICT-CONSTRAINT-CHAR is no longer significant but is still present for backward compatibility with existing metadata. It can be considered a hint for the client to adopt the following behavior, which is descriptive, not normative:

X: The client SHOULD NOT display the value to the user.

R: The client can display the value to the user but SHOULD NOT allow the user to change it.

W: The client can display the value and allow the user to change it.

Constraints on modification of metadata are now the responsibility of the server and are described in section 2.2.2.3.

```
METADICT-INT-VECTOR = / METADICT-INT-VECTOR SP INT

METADICT-STRING-VECTOR = / METADICT-STRING-VECTOR SP
                           METADICT-STRING-ITEM

METADICT-STRING-ITEM = *METADICT-STRING-ITEM-CHAR

METADICT-STRING-ITEM-CHAR = %x1-1F / %x21-5b / %x5d-ff ; unescaped
   / %x5c SP ; escaped space
   / %x5c %x5c ; escaped backslash
```

2.2.2.2.12 DOCINFO

Contains the service-relative URL of a **document** and its metadata.

```
DOCINFO = OBRACKET "document_name" VALSEP URL-STRING ARGSEP
   "meta_info" VALSEP METADICT CBRACKET
VECTOR-DOCINFO = OBRACKET STARTLIST 1*(DOCINFO LISTSEP) CBRACKET
```

A **DOCINFO** assumes that the URL specified by the *document_name* parameter is service-relative.

Following is an example encoded as sent over the wire.

```
%5bdocument%5fname%3dfolder1%2ffolder2%2fsmall%2etxt%3bmeta%5finfo%3d%5
bvti%5ftimelastmodified%3bSW%7c08+Jun+2006+21%3a40%3a07+%2d0000%5bvti%5
fmodifiedby%3bSW%7cuser%5fname%5bvti%5fauthor%3bSW%7cuser%5fname%5d%5d
```

Following is an example decoded for readability.

```
[document_name=folder1/folder2/small.txt;meta_info=[vti_timelastmodifie
d;SW|08 Jun 2006 21:40:07 -0000;vti_modifiedby;SW|user_name;
vti_author;SW|user_name]]
```

2.2.2.2.13 Document-List-Return-Type

Used by the server to return a **list** of **documents** and their metadata.

```
DOCUMENT-LIST-RETURN-TYPE = OBRACKET *(OBRACKET "document_name" VALSEP
   URL-STRING ARGSEP "meta_info" VALSEP METADICT CBRACKET) CBRACKET
```

2.2.2.2.14 Service-Return-Type

Used to return information about a **site**.

```
SERVER-RELATIVE-URL-STRING = URL-STRING
```

The URL MUST be service-relative.

```
SERVICE-RETURN-TYPE = OBRACKET "service name" VALSEP
   SERVER-RELATIVE-URL-STRING ARGSEP "meta_info" VALSEP METADICT CBRACKET
```

2.2.2.2.15 DOC-INFO-Request

Used to return information about a **document** name and its metadata.<3>

```
DOC-INFO-REQUEST = ARG-NAME DOCINFO
```

The **RPC-KEY-STRING** (section 2.2.2.2.5) in ARG-NAME MUST be "document". For more details see also **DOCINFO** (section 2.2.2.2.12).

2.2.2.2.16 URL-Directory

Provides the name and metadata associated with a given URL.

```
URL-DIRECTORY = OBRACKET "url" VALSEP URL-STRING ARGSEP "meta_info"
   VALSEP METADICT CBRACKET
VECTOR-URL-DIRECTORY = OBRACKET STARTLIST 1*(URL-DIRECTORY LISTSEP) CBRACKET
```

2.2.2.2.17 Status

Used to send back a status error code.

```
STATUS-CODE = UNSIGNED-INT
STATUS = OBRACKET "status" VALSEP STATUS-CODE ARGSEP "osstatus" VALSEP
   STATUS-CODE ARGSEP "msg" VALSEP STRING ARGSEP "osmsg" VALSEP STRING
   CBRACKET
```

2.2.2.2.17.1 ErrorCodes

Error ID / code	Description
V_AUTH_NOT_FOR_METHOD 0x000E0001	The current user is not authorized to execute this method.
V_AUTH_METHOD_UNKNOWN 0x000E0002	The method is not recognized.
V_AUTHORING_DISABLED 0x000E001A	Authoring is disabled for this server.
V_BAD_CHAR_SERVICE_NAME 0x0005000E	Invalid character in site name.
V_BAD_CHARS_IN_URL 0x00090070	The URL contains invalid characters.
V_BAD_FILETYPE 0x00090064	The file type being uploaded is blocked on this server.
V_BAD_URL 0x00090005	The provided URL is invalid.
V_CANT_COPY_FOLDER_WITH_SUBWEBS 0x00090046	A **folder** that contains **subsites** cannot be copied.
V_CANT_COPY_TO_SELF 0x00090025	A file cannot be copied onto itself.
V_CANT_DELETE_FOLDER_WITH_SUBWEBS 0x00090047	A folder that contains subsites cannot be deleted.
V_CANT_DELETE_SERVICE_WITH_SUBWEBS 0x00090044	A site with subsites cannot be deleted.
V_CANT_MOVE_THICKET_FOLDER 0x00090048	The specified file is a supporting file in a **thicket**, and so cannot be moved, renamed, deleted, or copied.
V_CANT_RENAME_FOLDER_WITH_SUBWEBS 0x00090045	A folder that contains subsites cannot be renamed.
V_CANT_RENAME_SERVICE_WITH_SUBWEBS 0x00090043	A site with subsites cannot be renamed.
V_CANT_RENAME_VDIR_SERVICE 0x00090042	The specified site cannot be renamed because it is mapped to a virtual directory in Internet Information Service (IIS).
V_CANT_REPARENT_SERVICE 0x00090041	Sites cannot be reparented as part of a rename operation.
V_CHECKOUT_REQUIRED 0x00090075	Files in this library require checkout before editing, and this file is not checked out.
V_CLOSE_FILE 0x00020006	The file could not be closed.
V_CLOSE_HANDLE_ERR	A handle could not be properly closed.

Error ID / code	Description
0x00030050	
V_CONFIG_ACCESS_ERROR 0x0003006B	General failure in accessing configuration information.
V_COPY_DIR 0x0002001C	Cannot copy the folder to the target folder.
V_COPY_FILE 0x00020055	Cannot copy file.
V_CREATE_DIRECTORY 0x00020003	The folder could not be created.
V_CREATE_FILE 0x00020005	The file could not be created.
V_DBW_NON_DBW_WEB 0x00110005	The supplied folder is the root content folder for a site; the protocol requires the folder to be opened using the http:// URL of the site.
V_DIR_ALREADY_EXISTS 0x0009000D	A folder with the specified name already exists.
V_DIR_GONE 0x0002001A	The specified folder does not exist.
V_DIRECTORY_ANON_UPLOAD_DISABLED 0x0002005A	Anonymous upload of files is not allowed for this folder.
V_DIRECTORY_ANON_UPLOAD_DISABLED_WEB_ROOT 0x0002005B	Anonymous uploads to the root of this site are not allowed.
V_DLL_ENTRY_NOT_FOUND 0x00020029	The specified entry point in the DLL could not be found.
V_DLL_OPEN_NUM 0x00020023	The provided DLL could not be opened.
V_DLL_OPEN_STR 0x00020024	The provided DLL could not be opened.
V_DLL_VERSION_INCOMPATIBLE 0x0002002A	The supplied DLL version is incompatible with the version of the server.
V_DNS_BAD_IP_ADDRESS 0x00130004	The IP address is invalid.
V_DNS_NO_RESOLVE_HOSTNAME 0x00130002	The host name could not be resolved.
V_DOC_CHECKED_OUT 0x0009000E	The file is currently locked for editing by another user.
V_DOC_COULD_NOT_PARSE 0x00100006	The file could not be processed by the smart parser.
V_DOC_IS_LOCKED	The specified file is currently in use.

Error ID / code	Description
0x00090040	
V_DOC_NOT_CHECKED_OUT 0x0009000F	The file is not checked out.
V_DOC_NOT_UNDER_SOURCE_CONTROL 0x00090011	The file is not under source control.
V_DOC_TIMESTAMP_MISMATCH 0x00090001	The server time stamp on the **document** does not match the client's time stamp for the document.
V_DOC_UNDER_SOURCE_CONTROL 0x00090010	The file is already under source control.
V_DOC_VERSIONING_NOT_SUPPORTED 0x0009003D	Versioning is not supported on this server; therefore, the request could not be completed.
V_DOC_WRONG_LOCK_TYPE 0x0009003C	The requested locking operation cannot be completed because the file is currently being edited by another user.
V_FILE_CANT_GET_TMP_DIR 0x0002002E	The temporary folder used on the server could not be accessed or found.
V_FILE_EMPTY_UPLOAD 0x0002005C	The file being uploaded is empty or does not exist.
V_FILE_EXISTS 0x00020011	The file could not be opened.
V_FILE_GONE 0x00020015	The file or folder could not be opened because it does not exist.
V_FILE_MAKE_HIDDEN_ERROR 0x00020051	The file or folder could not be marked as hidden.
V_FILE_MAKE_NOT_CONTENT_INDEXED_ERROR 0x00020050	An error occurred when attempting to mark the file as not indexable by search.
V_FILE_NOT_EXECUTE 0x00020025	The file could not be executed.
V_FILE_OPEN_FOR_READ 0x00020001	The file cannot be opened for reading.
V_FILE_OPEN_FOR_WRITE 0x00020002	The file cannot be opened for writing.
V_FILE_OPEN_READ_WRITE 0x00020010	The file could not be opened for reading and writing.
V_FILE_OUT_OF_DISK_SPACE 0x0002004D	Insufficient disk space to complete the operation.
V_FILE_QUOTA_EXCEEDED 0x00020058	The file size quota for this folder has been exceeded and the upload rejected.
V_FILE_QUOTA_UBANGEE	The file could not be saved because it exceeds the

Error ID / code	Description
0x00020059	maximum file size allowed on this site.
V_FILE_QUOTA_WARNING 0x00020057	The file size quota for this folder will soon be exceeded. Delete files in this folder to prevent uploads from failing because of quota issues.
V_FILE_RENAME_SRC_IN_USE 0x00020056	The file or folder could not be renamed because the file is in use.
V_FORMS_AUTH_NOT_BROWSER 0x000E0098	Authorization failed for this site because of a pluggable authentication provider. The protocol requires the user to log on to the user's authentication provider first, before accessing the library.
V_HTBL_CHANGE_WITHOUT_ROW_ID 0x000F0001	Can't change an HTML table row without a row identifier.
V_HTBL_DUPLICATE_ROW 0x000F0002	Row number already exists in this HTML table.
V_HTBL_BAD_FORMAT 0x000F0003	This HTML table is improperly formatted.
V_HTBL_ROW_NOT_FOUND 0x000F0004	Row number doesn't exist in this HTML table.
V_IIS_READ_LOCK_ERROR 0x00030057	A read lock for IIS cached information could not be acquired.
V_IIS_RESTART_SERVER_NEEDED 0x00030052	In order to complete installation of the components, a restart of IIS is required.
V_IIS_WRITE_LOCK_ERROR 0x00030058	A write lock for IIS cached information could not be acquired.
V_IMPERSONATE_LOGGED_ON_USER_ERR 0x0003004F	The application pool was unable to impersonate the user for the incoming request, and thus was unable to complete the request.
V_LOCK_FILE 0x0002000D	The file could not be locked. Usually returned because the file is already in use.
V_META_INFO_NOT_FOUND 0x0002003D	The meta information associated with the file could not be found.
V_NEED_TO_CREATE_FOLDER 0x00090023	The folder does not exist. The protocol requires that the folder be created before the operation can be completed.
V_NO_SOURCE_CONTROL 0x00090013	Source control is not functioning correctly.
V_NOT_DIR 0x0002001B	The specified URL is not a folder.
V_OFFNET_TOO_MANY_MINORVERSIONS 0x00210088	Cannot create another minor version.

Error ID / code	Description
V_OPEN_DIR_STREAM 0x00020009	The folder could not be opened.
V_OPEN_THREAD_TOKEN_ERR 0x00030051	A thread could not be created.
V_OVER_QUOTA 0x00090063	The changes could not be saved because the site has exceeded its quota.
V_OWSSVR_EMPTY_REQUIRED_FIELDS 0x00050086	Document checkin could not be completed because required metadata fields are missing.
V_OWSSVR_ERRORACCESSDENIED 0x001E0002	Access denied.
V_OWSSVR_ERRORHTTPACCESSFORBIDDEN 0x001E0009	Unable to access the server.
V_OWSSVR_ERRORHTTPUNAUTHORISED 0x001E0008	The current user does not have permissions to access any resources on this server.
V_OWSSVR_ERRORINCOMPDLLVER 0x001E0001	The server is running an incompatible version of core DLLs.
V_OWSSVR_ERRORSERVERERROR 0x001E0007	A general error has occurred on the server.
V_OWSSVR_ERRORSERVERINCAPABLE 0x001E0006	The server does not support this capability.
V_OWSSVR_ERRORSRVFILENOTFOUND 0x001E001D	The provided file or folder does not exist on this server.
V_PATH_NO_WINDOWS_DIR 0x0002002F	The server's user folder could not be found.
V_PATH_NO_WINDOWS_SYSTEM_DIR 0x00020030	The server's system folder could not be found.
V_PATH_NOT_FOUND 0x0002001D	The file or folder path was not found.
V_READ_FILE 0x0002000B	An error occurred while reading the file.
V_REG_EXP 0x0002000A	The regular expression was invalid.
V_REG_GET_SECURITY_ERROR 0x0003006F	Error reading security for a required registry key.
V_REG_SET_SECURITY_ERROR 0x00030070	Error setting security for a required registry key.
V_REMOVE_DIRECTORY 0x00020004	The folder could not be removed.

Error ID / code	Description
V_REMOVE_FILE 0x00020007	The file could not be removed.
V_RENAME 0x00020014	The file or folder could not be renamed for unspecified reasons.
V_RENAME_DEST_EXISTS 0x00020019	Cannot rename file or folder because the destination name already exists.
V_REVERT_TO_SELF_ERR 0x0003004E	The application pool was unable to revert to its native process identity and therefore was unable to complete the request.
V_RPC_CLIENT_TOO_OLD 0x0004000C	The version running on the server is too recent to be used with the client version.
V_SC_CHANGE_NOT_SUPPORTED 0x00090062	This version of the web server does not support changing source control settings.
V_SERVER_NO_CREATE_WEB 0x00030067	The web server does not support renaming or deleting subsites from client programs.
V_SERVICE_ANON_UPLOAD_DISABLED 0x00090051	Anonymous upload of files is not allowed for this site.
V_SERVICE_CANT_DELETE_WEB 0x00090052	The site cannot be deleted because the current user does not have administrator permissions to both the site and the parent site.
V_SERVICE_EXISTS 0x00050002	The website address is already in use.
V_SERVICE_RELOCK_TOPOLOGY_CHANGED 0x00090049	The operation failed because a subsite was created during the course of the operation.
V_SHTML_INTERPRETER_MODE_ERROR 0x00020045	The protocol requires that the HTML interpreter have execute permissions.
V_SHTML_INTERPRETER_NOT_FOUND 0x00020037	The server's HTML interpreting engine could not be found or loaded.
V_STAT_FILE 0x00020008	The status of the file could not be retrieved.
V_SVC_BAD_IPMASK 0x0009000B	The IP address mask provided is invalid because it contains spaces or other control characters.
V_SVC_BROWSER_RECALC_NO_META_FILE 0x0009001C	Failure recalculating links for the specified file.
V_SVC_BUSY 0x00090009	The web server is busy; try again later.
V_SYSERR_EXCEPTION_OCCURRED_AT 0x0008001A	System error: Exception occurred at a specific location in the code.
V_SYSERR_NT_EXCEPTION_ACCESS_VIOLATION	System error: Access violation.

Error ID / code	Description
0x0008000B	
V_SYSERR_NT_EXCEPTION_ARRAY_BOUNDS_EXCEEDED 0x0008000D	System error: Array bounds exceeded.
V_SYSERR_NT_EXCEPTION_DATATYPE_MISALIGNMENT 0x0008000C	System error: Data type misalignment.
V_SYSERR_NT_EXCEPTION_FLT_DENORMAL_OPERAND 0x0008000E	System error: Denormalized floating point operand.
V_SYSERR_NT_EXCEPTION_FLT_DIVIDE_BY_ZERO 0x0008000F	System error: Floating point divide by zero.
V_SYSERR_NT_EXCEPTION_FLT_INEXACT_RESULT 0x00080010	System error: Inexact floating point result.
V_SYSERR_NT_EXCEPTION_FLT_INVALID_OPERATION 0x00080011	System error: Invalid floating point operation.
V_SYSERR_NT_EXCEPTION_FLT_OVERFLOW 0x00080012	System error: Floating point overflow.
V_SYSERR_NT_EXCEPTION_FLT_STACK_CHECK 0x00080013	System error: Floating point stack check.
V_SYSERR_NT_EXCEPTION_FLT_UNDERFLOW 0x00080014	System error: Floating point underflow.
V_SYSERR_NT_EXCEPTION_INT_DIVIDE_BY_ZERO 0x00080015	System error: Integer divide by zero.
V_SYSERR_NT_EXCEPTION_INT_OVERFLOW 0x00080016	System error: Integer overflow.
V_SYSERR_NT_EXCEPTION_NONCONTINUABLE_EXCEPTION 0x00080018	System error: Attempt to continue after a non-continuable exception.
V_SYSERR_NT_EXCEPTION_PRIV_INSTRUCTION 0x00080017	System error: Attempt to execute a privileged instruction.
V_SYSERR_PREFIX 0x0008000A	General system exception encountered.
V_SYSERR_UNRECOGNIZED_EXCEPTION 0x00080019	System error: Unknown exception occurred.
V_THEME_ALREADY_EXISTS 0x0009002C	A theme with the specified name and version already exists on the server.
V_URL_DIR_NOT_FOUND 0x00090007	The folder that contains the URL specified could not be found within the site.
V_URL_NOT_FOUND 0x00090006	No file with the given URL could be found within the current site.
V_URL_TOO_LONG	The specified file or folder name is too long.

Error ID / code	Description
0x00090068	
V_URL_TOO_NESTED 0x0009000A	The URL provided has more than 32 directories.
V_UTIME_FILE 0x00020054	The modify time for the file could not be set.
V_WRITE_FILE 0x0002000C	An error occurred while writing the file.

2.2.2.2.18 Put-Option

Used to define the behavior of file upload operations.

```
PUT-OPTION-VAL = "atomic"
```

If this flag is specified, the server does all the needed checking to ensure that all the files can be updated before changing the first one. The server MAY<4> ignore this.

```
PUT-OPTION-VAL =/ "checkin"
```

The **document** is checked in after it is saved. This flag is used only to support long-term checkout operations. Servers MAY ignore this parameter if they choose not to support long-term checkout.

```
PUT-OPTION-VAL =/ "checkout"
```

Valid only if checkin is specified. Notifies the source control of the new content (checkin), but keeps the document checked out. This is the equivalent to checking in the document, and then checking it out again.

```
PUT-OPTION-VAL =/ "createdir"
```

The parent **folder** is created if it does not exist. By default, the server MUST require that the parent folder of a file or folder exist. If the client sends this option, the server MUST create the immediate parent folder of the file being created, if needed and if possible. For example, if folder1/folder2/file.txt is being created, the server MUST create folder2 if needed, but not folder1 if it does not already exist.

```
PUT-OPTION-VAL =/ "edit"
```

Uses the date and time that the document was last modified to determine whether the item has been concurrently modified by another user. This flag is used to prevent race conditions where two users could edit the same data. If this flag is specified and the inbound modification time does not match the value on the server, the server MUST reject the upload. The client SHOULD send this flag unless a higher level has indicated that it needs to overwrite changes.

```
PUT-OPTION-VAL =/ "forceversions"
```

Causes the server to act as though versioning is enabled, even if it is not. Servers MAY <5> ignore this parameter.

```
PUT-OPTION-VAL =/ "listthickets"
```

Requests that metadata be returned for **thicket** supporting files. The server MUST act as though this parameter was sent if the effective protocol version is less than 5.0.

```
PUT-OPTION-VAL =/ "migrationsemantics"
```

This option relaxes certain server-side checking during **site** migration operations. It allows clients to preserve certain metadata about who created the file and when, who last updated the file and when, and checkin comments. The server MAY ignore this option. If the server honors this option, it SHOULD require additional authorization of the caller and ignore the option if the authorization fails.<6>

```
PUT-OPTION-VAL =/ "noadd"
```

Does not add the document to source control. Clients that conform to the FrontPage Server Extensions: Website Management Protocol MUST NOT send this option. The server SHOULD ignore this option.

```
PUT-OPTION-VAL =/ "nounghost"
```

Does not update anything in the document content other than the **Web Parts**.

```
PUT-OPTION-VAL =/ "overwrite"
```

Uses the date and time that the document was last modified, as specified in the inbound metadata, rather than the extent of time on the server.

```
PUT-OPTION-VAL =/ "thicket"
```

Specifies that the associated file is a thicket supporting file. The server SHOULD detect that the upload includes a thicket that supports file and infer this flag.

```
PUT-OPTION = *(PUT-OPTION-VAL ",") PUT-OPTION-VAL
```

The PUT-OPTION data type MUST contain at least one PUT-OPTION-VAL.

2.2.2.2.19 Rename-Option

Used to define the behaviors of a rename operation.

```
RENAME-OPTION-VAL = "createdir"
```

Creates the parent **folder** if it does not already exist. This flag is analogous to the "createdir" PUT-OPTION-VAL (as specified in section 2.2.2.2.18) and has the same semantics.

```
RENAME-OPTION-VAL =/ "findbacklinks"
```

Requests that servers, implementing **link fixup**, fix the linked files other than those moved. The server MAY ignore this flag.

```
RENAME-OPTION-VAL =/ "nochangeall"
```

Do not perform link fixup on links in moved **documents**. This parameter is used in publishing scenarios. Clients that conform to the FrontPage Server Extensions: Website Management Protocol MUST NOT send this option. The server SHOULD ignore this option.

```
RENAME-OPTION-VAL =/ "patchprefix"
```

Simulates the move of a folder rather than a file. Clients that conform to the FrontPage Server Extensions: Website Management Protocol MUST NOT send this option; the server SHOULD ignore this flag for the usage defined in this protocol document.

```
RENAME-OPTION = "none"
   / RENAME-OPTION-VAL *("," RENAME-OPTION-VAL)
```

The client MUST send "none" if it does not specify any of the options given by a RENAME-OPTION-VAL.

2.2.2.2.20 Error-Option

Used to define the error-handling behavior of the method **setDocsMetaInfo** (section 3.1.5.3.39).

```
ERROR-OPTION = "keepGoing"
```

If this flag is specified, the server SHOULD continue attempting to apply metadata to **documents** even if errors occur.

```
ERROR-OPTION =/ "stopOnFirst"
```

If this flag is set, the server SHOULD stop processing on the first error that occurs.

2.2.2.2.21 Border-Specification

Used to indicate which borders are set for a document or as the default for a **service**.

```
BORDER-SPECIFICATION = BORDER-DEFAULT-OPTION
   / BORDER-NONE-OPTION
   / BORDER-OPTION *([","] BORDER-OPTION ) [":" BORDER-NAME]
```

The **BORDER-SPECIFICATION** appears in the **vti_borderaggregate** (section 2.2.2.3.14) and **vti_borderdefault** (section 2.2.2.3.15) metakeys and in the *border_spec* parameter of the **apply border** (section 3.1.5.3.3) method.

```
BORDER-DEFAULT-OPTION = "default"
```

If this flag is specified, the border is set to the default for the **site**.

```
BORDER-NONE-OPTION = "none"
```

If this flag is specified, no border is set.

```
BORDER-OPTION = "t" / "T"
```

If this flag is specified, the border is set for the top of the **page**.

```
BORDER-OPTION =/ "b" / "B"
```

If this flag is specified, the border is set for the bottom of the page.

```
BORDER-OPTION =/ "r" / "R"
```

If this flag is specified, the border is set for the right of the page.

```
BORDER-OPTION =/ "l" / "L"
```

If this flag is specified, the border is set for the left of the page.

```
BORDER-NAME = TOKEN
```

This value contains a name for the specified border.

2.2.2.2.22 Border-Aggregate-Specification

Used to communicate the border-specification applied to a **document**.

```
BORDER-AGGREGATE-SPECIFICATION = ACTUAL-BORDER-SPECIFICATION
   ["," SP VIRTUAL-BORDER-SPECIFICATION]
ACTUAL-BORDER-SPECIFICATION = BORDER-SPECIFICATION
```

```
VIRTUAL-BORDER-SPECIFICATION = BORDER-SPECIFICATION
```

A document's ACTUAL-BORDER-SPECIFICATION indicates the border that was set by default for the document or that has been specifically applied to the document. The VIRTUAL-BORDER-SPECIFICATION, if present, is the default border of the **site** that is used by the document when the ACTUAL-BORDER-SPECIFICATION is set to BORDER-DEFAULT-OPTION.

2.2.2.2.23 Theme-Parameters

Used to indicate which options are used when applying themes. The **THEME-PARAMETERS** contains an encoded value that records the choices for applying themes (see [MSDN-ThemeDef]) that use **cascading style sheets (CSS)**, color type, active graphics, and background type.

```
THEME-PARAMETERS = THEME-BACKGROUND THEME-ACTIVE-GRAPHICS
   THEME-VIVID-COLOR [THEME-USING-CSS]
THEME-BACKGROUND = "0" / "1"
THEME-ACTIVE-GRAPHICS = "0" / "1"
THEME-VIVID-COLOR = "0" / "1"
THEME-USING-CSS = "0" / "1"
```

If the THEME-BACKGROUND flag is specified as "0", the server MUST NOT use a graphic for the theme's **page** background. If the flag is specified as "1", the server MUST use a graphic for the theme's page background. If the THEME-ACTIVE-GRAPHICS flag is specified as "0", the server MUST use normal graphics for the theme. If the flag is specified as "1", the server MUST use active graphics for the theme.

If the THEME-VIVID-COLOR flag is specified as "0", the server MUST use normal colors for the theme. If the flag is specified as "1", the server MUST use vivid colors for the theme.

If the THEME-USING-CSS flag is specified as "0", the server MUST NOT use CSS to create the theme; the server SHOULD apply no theme, but MAY modify the HTML of the page to create the theme. If the flag is specified as "1", the server MUST use CSS to create the theme. If this flag is not specified, the server MUST default to "0".

2.2.2.2.24 Theme-Specification

Used to communicate the theme and theme-parameters applied to a **document** or **site**.

```
THEME-SPECIFICATION = THEME-NAME [SP THEME-PARAMETERS]
   / THEME-DEFAULT [SP THEME-PARAMETERS]
   / THEME-NONE
THEME-SPECIFIER = THEME-NONE / THEME-DEFAULT / THEME-NAME
THEME-NONE = "none"
THEME-DEFAULT = "default"
THEME-NAME = TOKEN
```

A **THEME-SPECIFICATION** of THEME-NONE indicates that a site or document has no applied theme. A **THEME-SPECIFICATION** of THEME-DEFAULT indicates that the document uses the site's default theme. A **THEME-SPECIFICATION** of THEME-NAME indicates that the site or document uses the named theme.

2.2.2.2.25 Theme-Aggregate-Specification

Used to communicate the aggregate theme-specification applied to a **document**.

```
THEME-AGGREGATE-SPECIFICATION = ACTUAL-THEME-SPECIFICATION
```

```
   ["," SP VIRTUAL-THEME-SPECIFICATION]
ACTUAL-THEME-SPECIFICATION = THEME-SPECIFICATION
VIRTUAL-THEME-SPECIFICATION = THEME-SPECIFICATION
```

A document's ACTUAL-THEME-SPECIFICATION indicates the theme that was set by default for the document or that has been specifically applied to the document. The VIRTUAL-THEME-SPECIFICATION, if present, is the site default theme that is used by the document when the ACTUAL-THEME-SPECIFICATION is set to THEME-DEFAULT.

2.2.2.2.26 Source-Control-Version

Used to communicate the version number of the source control system in use by a **site**.

```
SOURCE-CONTROL-VERSION = "V" SOURCE-CONTROL-VERSION-MAJOR
   "." SOURCE-CONTROL-VERSION-MINOR
SOURCE-CONTROL-VERSION-MAJOR = INT
SOURCE-CONTROL-VERSION-MINOR = INT
```

The SOURCE-CONTROL-VERSION-MAJOR and SOURCE-CONTROL-VERSION-MINOR numbers are the major and minor version numbers of the server software, respectively.

2.2.2.2.27 Source-Control-Document-Version

Used to communicate the version number of a **document** under source control.

```
SOURCE-CONTROL-DOCUMENT-VERSION = "V" DOCUMENT-VERSION-MAJOR
   ["." DOCUMENT-VERSION-MINOR]
DOCUMENT-VERSION-MAJOR = INT
DOCUMENT-VERSION-MINOR = INT
```

2.2.2.2.28 Element-ID

Used to communicate identifiers of nodes of the web structure of a **site**.

```
ELEMENT-ID = INT
TEMP-ELEMENT-ID = ELEMENT-ID ; MUST have a value > 0 and < 1000
PERMANENT-ELEMENT-ID = ELEMENT-ID ; MUST have a value > 1000
HOME-ELEMENT-ID = ELEMENT-ID ; MUST have a value = 1000
VECTOR-ELEMENT-ID = OBRACKET STARTLIST 1*( ELEMENT-ID LISTSEP) CBRACKET
```

Each element in a web structure for a site MUST have a unique **ELEMENT-ID**. When an element is created, it can have a temporary **ELEMENT-ID** sent as a TEMP-ELEMENT-ID, which the server MUST convert to a unique PERMANENT-ELEMENT-ID. The home page for the site has a special value for its **ELEMENT-ID** of HOME-ELEMENT-ID.

2.2.2.2.29 Element-Type

Used to communicate the type of object a node in the web structure of a **site** refers to.

```
ELEMENT-TYPE = "page" / "link"
```

Each element in a web structure has an **ELEMENT-TYPE** that specifies the type of object that the **STRUCTURE-ELEMENT** (section 2.2.2.2.32) refers to. A "page" refers to a static **document** within the site. A "link" is a reference to an external document.

2.2.2.2.30 Mode-Type

Used to update individual elements of the web structure of a **site**.

```
MOD-TYPE = "addNewPage"
```

This creates a new, empty **page** at the specified location and adds it to the web structure.

```
MOD-TYPE =/ "addExistingPage"
```

This adds an existing page to the web structure.

```
MOD-TYPE =/ "move"
```

This moves the navigation element for a page within the web structure.

```
MOD-TYPE =/ "delete"
```

This deletes the navigation element for a page from the web structure.

```
MOD-TYPE =/ "changeLabel"
```

This changes the label of the navigation element for a page in the web structure.

```
MOD-TYPE =/ "changeMetaInfo"
```

This merges the specified metadata with the existing metadata for the page in the web structure.

```
MOD-TYPE =/ "copyPage"
```

A copy of the specified page is made in the new location and added to the web structure. If the specified page does not exist, a new, empty page is created at the new location.

```
MOD-TYPE =/ "changeUrl"
```

The target of the navigation element link in the web structure is set to the specified location.

2.2.2.2.31 Nav-Key-Value

Used to communicate individual data elements of nodes of the web structure of a **site**.

```
NAV-KEY-VALUE-DTLP = [ARGSEP] "DTLP" VALSEP TIME
NAV-KEY-VALUE-EID = [ARGSEP] "eid" VALSEP ELEMENT-ID
NAV-KEY-VALUE-EID-CHILDREN = [ARGSEP] "eidChildren" VALSEP VECTOR-ELEMENT-ID
NAV-KEY-VALUE-EID-MOD = [ARGSEP] "eidMod" VALSEP ELEMENT-ID
NAV-KEY-VALUE-EID-PARENT = [ARGSEP] "eidParent" VALSEP ELEMENT-ID
NAV-KEY-VALUE-EID-REF = [ARGSEP] "eidRef" VALSEP ELEMENT-ID
NAV-KEY-VALUE-EID-TEMP = [ARGSEP] "eidTemp" VALSEP TEMP-ELEMENT-ID
NAV-KEY-VALUE-ELEMENT-TYPE = [ARGSEP] "eType" VALSEP ELEMENT-TYPE
NAV-KEY-VALUE-METAINFO = [ARGSEP] "meta-info" VALSEP METADICT
NAV-KEY-VALUE-MOD-TYPE = [ARGSEP] "mType" VALSEP MOD-TYPE
NAV-KEY-VALUE-NAME = [ARGSEP] "name" VALSEP STRING
NAV-KEY-VALUE-URL = [ARGSEP] "url" VALSEP SERVICE-RELATIVE-URL-STRING


NAV-KEY-VALUE = NAV-KEY-VALUE-DTLP / NAV-KEY-VALUE-EID / NAV-KEY-VALUE-EID-CHILDREN
   / NAV-KEY-VALUE-EID-MOD / NAV-KEY-VALUE-EID-PARENT / NAV-KEY-VALUE-EID-REF
   / NAV-KEY-VALUE-EID-TEMP / NAV-KEY-VALUE-ELEMENT-TYPE / NAV-KEY-VALUE-METAINFO
   / NAV-KEY-VALUE-MOD-TYPE / NAV-KEY-VALUE-NAME / NAV-KEY-VALUE-URL
```

The **NAV-KEY-VALUE** is an **RPCKEY and RPCVALUE** (section 2.2.2.2.5) pair which specifies web structure elements. The leading ARGSEP MUST be present in a **NAV-KEY-VALUE**, except in URL Mode when it is the first key after an OBRACKET or at the start of a response, in which case it MUST NOT be present.

2.2.2.2.32 Structure-Element

Used to communicate the web structure of a **site**.

```
ELEMENT-NAV-KEY-VALUE = NAV-KEY-VALUE-DTLP / NAV-KEY-VALUE-EID
   / NAV-KEY-VALUE-EID-CHILDREN / NAV-KEY-VALUE-EID-PARENT
   / NAV-KEY-VALUE-EID-TEMP / NAV-KEY-VALUE-ELEMENT-TYPE / NAV-KEY-VALUE-METAINFO
   / NAV-KEY-VALUE-NAME / NAV-KEY-VALUE-URL


STRUCTURE-ELEMENT = OBRACKET 1*(ELEMENT-NAV-KEY-VALUE) CBRACKET

VECTOR-STRUCTURE-ELEMENT = OBRACKET STARTLIST 1*(STRUCTURE-ELEMENT LISTSEP)
   CBRACKET
```

The web navigation structure is represented by a VECTOR-STRUCTURE-ELEMENT that contains a breadth-first traversal of the web structure navigation hierarchy.

2.2.2.2.33 Structure-Modification

Used to update the web structure of a **site**.

```
MOD-NAV-KEY-VALUE = NAV-KEY-VALUE-DTLP
   / NAV-KEY-VALUE-EID-MOD / NAV-KEY-VALUE-EID-PARENT / NAV-KEY-VALUE-EID-REF
   / NAV-KEY-VALUE-ELEMENT-TYPE / NAV-KEY-VALUE-METAINFO
```

```
   / NAV-KEY-VALUE-MOD-TYPE / NAV-KEY-VALUE-NAME / NAV-KEY-VALUE-URL

STRUCTURE-MODIFICATION = OBRACKET 1*(MOD-NAV-KEY-VALUE) CBRACKET

VECTOR-STRUCTURE-MODIFICATION = OBRACKET STARTLIST 1*(STRUCTURE-MODIFICATION
   LISTSEP) CBRACKET
```

Each element in a **STRUCTURE-MODIFICATION** has an "eidMod" **ELEMENT-ID** (section 2.2.2.2.28) to identify the element being operated on, an **ELEMENT-TYPE** (section 2.2.2.2.29) that specifies the type of object that the **STRUCTURE-MODIFICATION** refers to, and a **MOD-TYPE** (section 2.2.2.2.30) that specifies the type of change to be made to the web structure.

2.2.2.2.34 Web-Navigation-URL

Used to communicate the links within a **document** to web navigation structure elements.

```
WEB-NAVIGATION-URL = SEQUENCE-URL / BACK-NEXT-NAVIGATION-URL
SEQUENCE-URL = "S|" STRUCTURE-ELEMENT-URL
BACK-NEXT-NAVIGATION-URL = "B|" STRUCTURE-ELEMENT-URL
STRUCTURE-ELEMENT-URL = "sid:" ELEMENT-ID
```

2.2.2.2.35 Linkinfo-Item

Used to communicate information about the HTML links found in a **document**. Links are characterized by their target status, their type or source within a document, the security of the transport used, and whether they link to a static or dynamic **page**.

```
LINKINFO-ITEM = LINKINFO-CODE "|" LINKINFO-TARGET
LINKINFO-TARGET = ABSOLUTE-URL / SERVICE-RELATIVE-URL
LINKINFO-CODE = LINKINFO-STATUS LINKINFO-TYPE LINKINFO-SECURITY
   LINKINFO-DYNAMICITY
LINKINFO-STATUS = "N" / "D" / "F" / "W" / "U"
LINKINFO-TYPE = "A" / "B" / "C" / "D" / "E" / "F" / "G" / "H"
   / "I" / "J" / "K"  / "L" / "M" / "N" / "O" / "P" / "Q"
   / "R" / "S" / "T" / "U" / "V" / "X" / "Y" / "Z"
LINKINFO-SECURITY = "H" / "S" / "T" / "U"
LINKINFO-DYNAMICITY = "D" / "G" / "H" / "L" / "S "
```

The LINKINFO-STATUS refers to the target of the link and is encoded with one of the following letter values.

Value	Meaning
N	The link is identified as broken.
D	The link is to a **folder** without a welcome page.
F	The link is to a file.
W	The link is to a folder with a welcome page.
U	The link status is unknown.

The LINKINFO-TYPE or source of a link is encoded by one of the following letters.

Value	Meaning
A	The link is within the ACTION attribute of a FORM tag.
B	The link has been created by a **bot**.
C	The link is for a page hit bot.
D	The link is within script or within an OBJECT tag's CLASSID, PROGID, or CODEBASE attribute.
E, Q, Y	The link is within a STYLESHEET attribute or an include of a CSS.
F	The link is within a FRAME tag.
G	The link is to a templated document.
H	The link is an HREF. This is the default type for a link.
I	The link is a bot include directive.
J	The link is a database field.
K	The link is a bookmark.
L	The link is a target in an HTML image map generated from an image map bot.
M	The link is in an OBJECT tag's USEMAP attribute.
N	The link is a Navigation link. This is the same as an HREF.
O,P,Z	The link is generated by a **Web Part**.
R	The link is within an ASP.NET page.
S	The link is within an SRC attribute or similar attribute of many tag types.
T	The link is to the index file used by a text search bot on this page.
U	The link type is unknown.
V	The link is within database metadata.
X	The link is within XML.

The LINKINFO-SECURITY flag is encoded with one of the following letters.

Value	Meaning
H	The link is to an "HTTP:" URL.
T	The link is to an "SHTTP:" URL.
S	The link is to an "HTTPS:" URL.
U	The link transport security is unknown.

The LINKINFO-DYNAMICITY flag is encoded with one of the following letters.

Value	Meaning
D	The link is to a dynamic URL, such as a DLL.
G	A nonabsolute link from a templated document which does not fall into any other category.

Value	Meaning
H	The URL is a history link, containing a path segment with the string "_vti_history".
L	The URL is to a layouts page, containing a path segment with the string "_layouts".
S	The link is to a static URL. This is the default value.

2.2.2.2.36 Apply-Option

Used for the argument values in the *apply_opt* parameter of several methods.

```
APPLY-OPTION =  APPLY-OPT ["," APPLY-OPT ["," APPLY-OPT]]
APPLY-OPT = APPLY-OPT-WEB / APPLY-OPT-PAGE / APPLY-OPT-RFI
APPLY-OPT-WEB = "web"
APPLY-OPT-PAGE = "page"
APPLY-OPT-RFI = "rfi"
```

2.2.2.3 Metadata

Files, **folders**, and **sites** in servers have an associated metadictionary, which contains strings (called keys or metakeys) that are mapped to strongly typed values. These metakey-value pairs are called metadata. Server implementations use the metadictionary to store details about entities for later use by the server. Clients store values in metadictionaries locally or on the server for later use by the same client or other clients. A limited number of well-known metakeys is used for client/server communication. These shared metakeys are described in this document.

2.2.2.3.1 Type

Each metakey listed has an associated value type, which is one of the **METADICT-VALUE** types specified in section 2.2.2.2.11. These are generic types which can be further specified in the description of each individual metakey.

2.2.2.3.2 Client Access

The Client Access heading refers to whether the client is able to set this metadata on the server.

Read-only: Some communication from server to client is based on configuration information and **site** settings or **document** information which is parsed and returned to the client; the client cannot change this information. These metakeys are identified as Read-only. Clients MUST NOT change the Read-only metakeys.

Read-write: Metadata that the client is able to set on the server is identified as Read-write.

2.2.2.3.3 Applies To

Metadata is associated with various entities on the server, which are identified in the Applies To heading.

Service: The **Service** value refers to **service** metadata associated with the server or a particular **site**.

Folder: The **Folder** value refers to metadata associated with a **folder**, **document library**, or **list** container.

File: The **File** value refers to metadata associated with a file or **document**.

2.2.2.3.4 vti_adminurl

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_adminurl** metakey contains the value of the URL used for administration of the site.

The client MAY display this **page** in response to a user request made in its user interface.

The server MUST return the URL of the administration page for this site to ensure that the client invokes the correct page through its user interface.<7>

The server MUST support password administration through a page accessed by appending the string "?page=security.htm" to the value returned in this metakey. The client SHOULD expose a user interface option to change passwords that calls the page created by concatenating this string to the value in this metakey.

2.2.2.3.5 vti_approvaldate

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File

The **vti_approvaldate** metakey is the most recent time that the document review status stored in **vti_approvallevel** (section 2.2.2.3.6) was changed. This value is empty if the **vti_approvallevel** metakey is empty.

The server MUST store the current time in this metakey when the client changes the **vti_approvallevel** value to a nonempty string. The server MUST clear this metakey when the **vti_approvallevel** value is set to an empty string.

2.2.2.3.6 vti_approvallevel

Attribute	Value
Type	STRING (section 2.2.2.1.5)
Client Access	Read-Write
Applies to	File

The **vti_approvallevel** metakey is used to indicate which, if any, of the approval level values stored in the **vti_approvallevels** (section 2.2.2.3.7) collection applies to the document.

The server MUST store this as the value set by the client. The client MAY set this value for a document, and if it does, it MUST be a value from the collection of possible values returned by the **vti_approvallevels** metakey.

2.2.2.3.7 vti_approvallevels

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-write
Applies to	Service

The **vti_approvallevels** metakey is a list of the categories available for application to the **vti_approvallevel** (section 2.2.2.3.6) for a **document**.

A client MAY allow the user to change the values available for document approval levels.

The server MUST accept client updates to the vti_approvallevels available on the service. The server SHOULD default to the following list if the client has not updated this metakey:

- Approved
- Denied
- Pending Review

Following is an example:

```
[vti_approvallevels; VW|Approved Denied Pending\\ Review]
```

The double backslash marks an escaped backslash in the **METADICT-STRING-VECTOR** (section 2.2.2.2.11).

2.2.2.3.8 vti_approvedby

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_approvedby** metakey is used to store the login name of the client user who sets the **vti_approvallevel** (section 2.2.2.3.6) value.

If the **vti_approvallevel** metakey is not empty, the server MUST store the login username used to change the value of the **vti_approvallevel** metakey. The server MUST clear this metakey if the **vti_approvallevel** metakey value is empty.

2.2.2.3.9 vti_assignedby

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_assignedby** metakey is the username associated with the client that changed the **vti_assignedto** (section 2.2.2.3.11) value for the **document**. Contains no value if the document has no **vti_assignedto** value.

If the **vti_assignedto** metakey is not empty, the server MUST store the login username used to change the value of the **vti_assignedto** metakey. The server MUST clear this metakey if the **vti_assignedto** metakey value is empty.

2.2.2.3.10 vti_assigneddate

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File

The **vti_assigneddate** metakey is the time that the most recent nonempty change to the **vti_assignedto** (section 2.2.2.3.11) metakey was made. Contains no value if the **vti_assignedto** value is empty.

The server MUST store the time at which the client changed the value of the **vti_assignedto** metakey if the **vti_assignedto** metakey is not empty. The server MUST clear this metakey if the **vti_assignedto** metakey value is empty.

2.2.2.3.11 vti_assignedto

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	File

The **vti_assignedto** metakey is the username the **document** is assigned to in the client, if any.

The server MUST store this value as set by the client and update the values of **vti_assignedby** (section 2.2.2.3.9) and **vti_assigneddate** (section 2.2.2.3.10) at the same time. The client MAY set this value for a document, and MAY set it to an empty string.

2.2.2.3.12 vti_author

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_author** metakey is the username of the client user that first puts this **document** on the server.

The server MUST store the login username associated with the client used to put this document on the server in this metakey when the document is initially created on the server.

2.2.2.3.13 vti_backlinkinfo

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)

Attribute	Value
Client Access	Read-only
Applies to	File

The **vti_backlinkinfo** metakey is a list of **URL-STRING** (section 2.2.2.2.3) values specifying the service-relative URLs of the documents which link to this document.

The server MUST maintain metadata about the links in each document in the **service** in its **vti_linkinfo** (section 2.2.2.3.47) metakey, and examine these to create a list of the other documents in the site that link to this document to set the contents of this metakey.

2.2.2.3.14 vti_borderaggregate

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_borderaggregate** metakey is a **BORDER-AGGREGATE-SPECIFICATION** (section 2.2.2.2.22) specifying the borders in use in the **page**. This metakey is used for pages with both borders and navigation bars.

The server MUST obtain this string by parsing the **document** for a META element tag with a NAME attribute of "Microsoft Border" and returning the value of the CONTENT attribute. The server MAY cache this value for return to the client on request.

The client MUST use the **apply border** (section 3.1.5.3.3) method to change this value for the document.

Following is an example:

```
[vti_borderaggregate;SR|default]
[vti_borderaggregate;SR|tlb:darkborder]
```

2.2.2.3.15 vti_borderdefault

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_borderdefault** metakey contains a **BORDER-SPECIFICATION** (section 2.2.2.2.21) with the default border settings on the **site**.

The server MUST apply this default to all **documents** that do not have their own border settings.

The server MUST maintain this information for return to the client upon request. If no value has been set for this metakey, the server SHOULD default to a value of BORDER-OPTION-NONE in the **BORDER-SPECIFICATION** (section 2.2.2.2.21).

This value cannot be set by the client directly, but MAY be set using the **apply border** (section 3.1.5.3.3) method.

Following is an example:

```
[vti_borderdefault;SR|tb]
```

2.2.2.3.16 vti_cachedbodystyle

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_cachedbodystyle** metakey contains the opening BODY tag of the **document**, which can contain style information, such as bgcolor, background, or bgproperties attributes.

The server MAY parse the document for the BODY tag and return its contents on request. The client can only set this value by changing the document.

Following is an example:

```
[vti_cachedbodystyle;SR|<BODY bgColor=transparent>]
```

2.2.2.3.17 vti_candeleteversion

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	File

The **vti_candeleteversion** metakey contains a flag indicating whether the client has sufficient permissions to delete versions of the **document** under source control.

The server MUST check permissions and return this calculated value to the client when requested.

2.2.2.3.18 vti_canmaybeedit

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

The **vti_canmaybeedit** metakey contains a flag indicating whether the client user has sufficient permissions to edit items in the **list** folder. Individual **documents** MAY have different permission levels applied.

The server MUST check permissions and return this calculated value to the client when requested.

2.2.2.3.19 vti_cannotlisturls

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_cannotlisturls** metakey contains an INT flag indicating whether the client user has sufficient permissions to browse **folders** on the server. A value of 0 indicates that the client can browse folders on the server. A value of 1 indicates that the client does not have sufficient permissions.

The server MAY check user permissions and return this calculated value to the client when requested.

2.2.2.3.20 vti_casesensitiveurls

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_casesensitiveurls** metakey contains an INT flag indicating whether the server is case-insensitive with respect to URLs. If the value is 0 (the default) the server is case-insensitive with respect to URLs. If the value is 1, the server is case-sensitive with respect to URLs.

The server SHOULD include this key as an INT in the metadata returned by the **open service** (section 3.1.5.3.24) method.

The server MUST return 0 or 1 for this metakey. It MUST be 0 if URLs that differ only by case are considered equivalent. The client SHOULD assume that the value is 0 if this key is not present.

Following is an example:

```
[vti_casesensitiveurls;IX|0]
```

2.2.2.3.21 vti_categories

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-write
Applies to	Service, File

When referring to a **document**, the **vti_categories** metakey contains the list of categories which have been applied to the document.

When referring to a **site**, this metakey contains a list of categories which can be applied to documents on the site.

The client MAY update the list of categories for a site on the server. The server MUST accept updates from the client for a site. The client MAY update the list of categories applied to a document. The server MUST accept updates from the client for the document. If the list of categories applied to the

document includes categories that do not appear in the list of categories for the site, the server SHOULD update the category list on the site to include the new categories.

The server SHOULD<8> provide a default list of categories for a site.

Following is an example:

```
[vti_categories;VR|Travel Expense\\ Report Business]
```

2.2.2.3.22 vti_charset

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_charset** metakey contains a string containing the name of the character set used by the **document**, as defined in [RFC2616] section 3.4. This is the name found in the CHARSET parameter of the document's Content-Type header, or in the CONTENT attribute of the META element tag with an HTTP-EQUIV attribute of "Content-Type" (case is not significant) in the header of the document, if any.

The server MUST determine the character set for the document (if known), and return this value to the client. The client can only set this value by changing the document.

Following is an example:

```
[vti_charset;SR|ISO-8859-1]
```

2.2.2.3.23 vti_custommasterurl

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	Service

The **vti_custommasterurl** metakey is the **server-relative URL** for a custom master web page. The site MAY specify a customized master web page to be applied within ASP.NET code pages (.aspx files) using the ~master/custom.master token. This token will be expanded to the URL contained in this metakey value.

The client MAY set this value for a site. The server MUST store and return this value on request.

2.2.2.3.24 vti_defaultcharset

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	Service

The **vti_defaultcharset** metakey contains the name of the default character set used by the **site**. This is a charset value as defined in [RFC2616] section 3.4.

Clients MAY set this value for a site. The server MUST accept a change to this value and use this value for **documents** which do not have a character set defined.

The server MUST accept the following values for the default character set. The server MAY accept synonyms for the following values, and MAY accept additional character sets:

Value	Value	Value
big5	iso-8859-8	windows-1250
euc-jp	iso-8859-9	windows-1251
euc-kr	iso-8859-10	windows-1252
gb2312	iso-8859-15	windows-1253
gb18030	koi8-r	windows-1254
iso-2022-jp	ks_c_5601-1987	windows-1255
iso-8859-1	shift_jis	windows-1256
iso-8859-2	**Unicode**	windows-1257
iso-8859-4	unicodeFFFE	windows-1258
iso-8859-5	us-ascii	x-undefined
iso-8859-6	**UTF-8**	
iso-8859-7	windows-874	

Following is an example:

```
[vti_defaultcharset;SR|windows-1252]
```

2.2.2.3.25 vti_defaultlanguage

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	Service

The **vti_defaultlanguage** metakey is the default language in use by the **site**. This is a language-code as specified in [RFC2616] section 3.10.

Clients MAY set this value for a site. The server SHOULD accept a change to this value and use this value for **documents** that do not have a character set defined. The server MAY limit the set of values to which this metakey MAY be changed.

Following is an example:

```
[vti_defaultlanguage;SW|en-us]
```

2.2.2.3.26 vti_description

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	File

The **vti_description** metakey contains the comments associated with the **document** by the client.

The client MAY set this value in a request to the server. The server MUST store and return this value on request.

Following is an example:

```
[vti_description;SW|My favorite ice-cream flavors]
```

2.2.2.3.27 vti_dirlateststamp

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	Folder

The **vti_dirlateststamp** metakey is a time stamp that records the approximate time of the first client call to the **list documents** (section 3.1.5.3.20) or **move document** (section 3.1.5.3.23) methods which included this folder, following a change to the folder contents or metadata.

The server SHOULD include this key in folder metadata it returns to the client.

If the client caches the response of the **list documents** (section 3.1.5.3.20) method requests, it SHOULD cache this time stamp, and send this value in the *folderList* parameter in subsequent calls to the **list documents** method. Servers SHOULD use the value during processing of a call to the **list documents** method for optimization, to only return data for folders which are out of date on the client.

Following is an example:

```
[vti_dirlateststamp;TX|08+Jan+2000+19:09:27+-0000]
```

2.2.2.3.28 vti_disablewebdesignfeatures

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_disablewebdesignfeatures** metakey contains a string used by the server to disable web design features by obsolete clients.<9>

The server SHOULD set this metakey to the default value "wdfopensite" to prevent editing by incompatible obsolete clients. A client implementing this protocol SHOULD ignore this metakey.

2.2.2.3.29 vti_disablewebdesignfeatures2

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_disablewebdesignfeatures2** metakey is a list of string tokens used by the server to indicate which web design features are disabled.

The server MUST send this list on request by the client. The client SHOULD disable the listed editing features in its user interface.

The following tokens MAY be set in this metakey.

Value	Meaning
wdfbackup	Disable web backup.
wdfrestore	Disable web restore.
wdfpackageimport	Disable web package import.
wdfpackageexport	Disable web package export.
wdfthemeweb	Disable theme support for the **site**.
wdfthemepage	Disable theme support for individual pages.
wdfnavigationbars	Disable support for navigation bars.
wdfnavigationview	Disable the Navigation view for this site.
wdfpublishview	Disable the Remote site view for this site.
wdfpublishselectedfile	Do not allow the selected file to be published.
wdfopensite	Disable access to the entire site.
wdfnewsubsite	Do not allow the creation of a new **subsite**.

Following is an example:

```
[vti_disablewebdesignfeatures2;VR|wdfthemeweb wdfthemepage]
```

2.2.2.3.30 vti_doclibwebviewenabled

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-write
Applies to	Service

The **vti_doclibwebviewenabled** metakey specifies whether to display a webpage view of available **document libraries** to clients which open **documents** from or save documents to the **site**. A value of 0 means the webpage view is not enabled. A value of 1 means the webpage view is enabled.

The server MUST ignore this setting.

2.2.2.3.31 vti_donotpublish

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-write
Applies to	File

The **vti_donotpublish** metakey specifies whether to not publish this **page**. If the value is true, the page is not published. If the value is false, the page is published. Publish operations are performed by the client; this setting has no effect on server behavior.

Clients SHOULD set this metakey for **documents** based on user choice in the client user interface. The server MUST store and return this value on client request.

2.2.2.3.32 vti_etag

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_etag** metakey contains an HTTP Etag for the **document** as specified in [RFC2616] section 14.19.

The server MUST create this value as needed and return this value on request by the client.

2.2.2.3.33 vti_featurelist

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_featurelist** metakey is a list of features that MAY be supported by this version of the server software, but are not supported by this particular **site**.

The server MUST send this list on request by the client. The client SHOULD disable the listed features in its user interface.

The list of possible features are divided into the following four sections: Server Features, Service Features, Access Control Features, and Document Features.

Server Features

Value	Meaning
vti_ServerEmailTransport	Server supports email transport.
vti_ServerIndexServer	Server has Index server running.
vti_ServerODBC	Server supports Open Database Connectivity (ODBC).
vti_ServerASP	Server supports Active Server Pages (ASP).
vti_TimedDocEvents	Server supports a timer service for rules.

Service Features

Value	Meaning
vti_ServiceRename	Can rename a site.
vti_ServiceRemove	Can remove a site.
vti_ServiceMarkUrlDirExec	Can make a URL **folder** executable.
vti_ServiceMarkUrlDirBrowse	Can make a URL folder browseable.
vti_ServiceStructureStore	Can store and read navigational structure information.
vti_ServiceThemes	Can list and apply themes.
vti_ServiceMarkUrlDirScript	Can mark a URL folder scriptable.

Access Control Features

Value	Meaning
vti_ACAll	All access controls possible.
vti_ACRegisteredEndUsers	Restrict access to only registered users.
vti_ACIPAddresses	Set access by IP address.
vti_ACCreateNewUsers	Can create new users.
vti_ACChangePassword	Can change current user's password.
vti_ACGroups	Can perform all operations on groups.
vti_ACModifyGroups	Can change members of group.
vti_ACCreateNewGroups	Can create new groups.
vti_ACUseDomains	**Domain** use supported.
vti_AC20	Microsoft FrontPage 2.0 style access control.
vti_ACNoUserGroup	Internal key (no defined use).

Document Features

Value	Meaning
vti_DocSaveToDB	Save to database enabled from within **documents**.

Following is an example:

```
[vti_featurelist;VX|vti_ACAll vti_ServerEmailTransport
vti_ServerIndexServer vti_ServerODBC vti_ServerASP
vti RulesScript vti TimedDocEvents vti ServiceMarkUrlDirExec
vti_ServiceMarkUrlDirBrowse vti_ServiceMarkUrlDirScript
vti DocSaveToDB]
```

2.2.2.3.34 vti_filesize

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	File

The **vti_filesize** metakey is the size of the **document** in bytes.

The server MUST determine the size of the file, in bytes, and return this value on request by the client.

Following is an example:

```
[vti_filesize;IX|1120]
```

2.2.2.3.35 vti_generator

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_generator** metakey identifies the name and version of the HTML authoring tool or application that created the **document**, if it can be determined by parsing the document for the CONTENT attribute of a META element tag containing a NAME attribute of "generator" (case is insignificant).

The server MUST parse the document for this value. The server MAY cache the value, and MUST send this value on request by the client.

2.2.2.3.36 vti_hasdefaultcontent

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	File

The **vti_hasdefaultcontent** metakey specifies whether the **document** is a templated document.

The server SHOULD maintain this value for templated documents if it supports templated documents. The server SHOULD return true if the document is a templated document on the server. The server SHOULD return false if it is an untemplated document.

2.2.2.3.37 vti_hassubdirs

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

The **vti_hassubdirs** metakey specifies whether the folder has subdirectories.

The server SHOULD return this key and set it to true only if the folder has subdirectories. The client MAY use this key to decide whether to display user interface elements to expand a node in a rendered folder hierarchy.

Following is an example:

```
[vti_hassubdirs;BR|true]
```

2.2.2.3.38 vti_htmlextensions

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_htmlextensions** metakey lists the file extensions of the types of web pages supported by the server.

Each file extension is separated with a period (.) and the string begins and ends with a period (.) if there are one or more entries.

The server MUST maintain this value and send it on client request.

Following is an example:

```
[vti_htmlextensions;SX|.html.htm.shtml.shtm.stm.htt.htx.asp.aspx.alx]
```

2.2.2.3.39 vti_httpdversion

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_httpdversion** metakey specifies the server name and version number as passed by the Common Gateway Interface (CGI) environment variable, SERVER_VERSION.

Following is an example:

```
[vti_httpdversion;SX|Microsoft-IIS/6.0]
```

2.2.2.3.40 vti_ignorekeyboard

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-write
Applies to	Service

The **vti_ignorekeyboard** metakey is a flag specifying whether the client considers the keyboard language setting when determining the language and encoding of newly created pages.

This metakey is set and used by the client. The server MUST store and return this metakey to the client on request. Servers MUST NOT make use of this metakey.

2.2.2.3.41 vti_isbrowsable

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

True if users can browse files in the current directory; otherwise, false. The server SHOULD return this key and set its value to TRUE only if the folder contents are accessible through normal HTTP requests. The server MUST use and maintain its own value rather than use a client-supplied value for this key.

Example:

```
[vti_isbrowsable;BR|false]
[vti_isbrowsable;BR|true]
```

2.2.2.3.42 vti_ischildweb

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

The **vti_ischildweb** metakey specifies whether the folder is the root of another **site** (a **subsite**) within this site.

The server SHOULD include this key on folder metadata it enumerates when the folder is the root of another **service**. The client MAY use this information to avoid further calls to the **url to web url** (section 3.1.5.3.41) method when traversing a folder hierarchy that might span services. The client also MAY use this key to indicate to the user that the folder represents a service boundary.

Following is an example:

```
[vti_ischildweb;BR|true]
```

2.2.2.3.43 vti_isexecutable

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

The vti_isexecutable metakey specifies whether the folder is executable. The server SHOULD include this BOOLEAN key on folder metadata. If the value is TRUE, the server MUST permit execution of programs in the folder. The server MUST use and maintain its own value rather than use a client-supplied value for this key.

Example:

```
[vti_isexecutable;BR|false]
```

2.2.2.3.44 vti_isscriptable

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	Folder

The **vti_isscriptable** metakey specifies whether the folder is scriptable. The server SHOULD include this BOOLEAN key on folder metadata. If the value is TRUE, the file or the contents of the folder SHOULD be served, even if they are script files. If FALSE, the server SHOULD only allow static files to be served. The server MUST use and maintain its own value rather than use a client-supplied value for this key.

Example:

```
[vti_isscriptable;BR|false]
[vti_isscriptable;BR|true]
```

2.2.2.3.45 vti_language

Attribute	Value
Type	**STRING** (section 2.2.2.1.5), **INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	File, Service

The **vti_language** metakey has different content depending on the context where it appears.

When referring to a **document**, this metakey is the language set in the META element tags for the document, in the format of a language-tag as specified in [RFC2616] section 3.10.

The server MUST parse the document to obtain the value for this metakey, and SHOULD cache it for return to the client on request. This value is specified in the document in the CONTENT attribute of a META element tag with an HTTP-EQUIV attribute value of "Content-Language" (case is not significant). The client cannot set this value directly, but can change it by updating the document.

When referring to a **site**, this metakey is the **language code identifier (LCID)** in use for the site. The server MUST supply this configuration value to the client on request.

Following is an example for a document:

```
[vti_language;SR|en-us]
```

Following is an example for a site:

```
[vti_language;IR|1033]
```

2.2.2.3.46 vti_linkbars

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	File

The **vti_linkbars** metakey specifies the web structure links that appear in this **document**. This consists of a list of WEB-NAVIGATION-URLs to the documents or links represented by the **ELEMENT-IDs** (section 2.2.2.2.28) in the web structure.

The server MUST maintain this metakey based on the web structure data for each document. The client cannot set this value directly, but MAY change it by updating the web structure with the **put web struct** (section 3.1.5.3.29) method or the **replace web struct** (section 3.1.5.3.35) method.

Following is an example:

```
[vti_linkbars;VR|S|sid:1002 S|sid:1003]
```

2.2.2.3.47 vti_linkinfo

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	File

The **vti_linkinfo** metakey lists the value of each link on the current **page** along with information about the status, type, security, and dynamicity of each link, encoded as a **LINKINFO-ITEM** (section 2.2.2.2.35).

The server MUST parse the document for this metakey value and MAY cache the result for return to the client on request. The server MAY return an empty metakey if there are no links in the document.

Following is an example:

```
[vti_linkinfo;VX|FHUS|DEVGRP/prod1.htm FHUS|DEVGRP/prod2.htm]
```

2.2.2.3.48 vti_listbasetype

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Folder

The **vti_listbasetype** metakey specifies which of several supported base **list** types is used for the list associated with this folder.

This MAY be used by clients along with the value in **vti_listservertemplate** (section 2.2.2.3.54) to select an appropriate icon for the folder when displaying it in a user interface.

The following base List types are defined for the FrontPage Server Extensions: Website Management Protocol.

Value	Meaning
0	Generic List
1	**document library**
3	Discussion
4	Survey
5	Issue

Other values are not in use by the protocol. The server MUST return a value from this table to the client on request. This value MUST be set by the server when a List is created, and cannot be changed by the client.

2.2.2.3.49 vti_listenableminorversions

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	**Folder**

The **vti_listenableminorversions** metakey contains a flag indicating that minor version numbering is enabled for this **list**.

This flag, along with the flag value in **vti_listenableversioning** (section 2.2.2.3.51), SHOULD be used by the client to determine whether to enable the versioning user interface for this List.

2.2.2.3.50 vti_listenablemoderation

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)

Attribute	Value
Client Access	Read-only
Applies to	Folder

The **vti_listenablemoderation** metakey contains a flag indicating that this folder is associated with a **document library** that has enabled content approval.

If the server implements document library functionality, it MUST return this flag.

2.2.2.3.51 vti_listenableversioning

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	**Folder**

The **vti_listenableversioning** metakey contains a flag indicating that version numbering is enabled for this **list**.

This flag along with the flag value in **vti_listenableminorversions** (section 2.2.2.3.49) SHOULD be used by the client to determine whether to enable the versioning user interface for this list.

2.2.2.3.52 vti_listname

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Folder

The **vti_listname** metakey contains the **field internal name**, a **GUID** ([MS-DTYP] section 2.3.2.3) of the **list** bound to this folder.

2.2.2.3.53 vti_listrequirecheckout

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	**Folder**

The **vti_listrequirecheckout** metakey contains a flag that indicates whether source control is enabled for **documents** in the **document library** bound to this folder.

If the flag value is TRUE, documents MUST be checked out by the client to edit the document. Clients that do not check out the document MAY obtain a read-only copy of the document.

2.2.2.3.54 vti_listservertemplate

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	**Folder**

The **vti_listservertemplate** metakey contains an INT that indicates which **list template** is used for the **list** associated with this folder.

The client MAY use this value to display different user interface icons for folders based on different List templates, such as the Image Library template or the Workflow template.

The following List templates are defined for the FrontPage Server Extensions: Website Management Protocol.

Value	Meaning
100	Generic List Template
101	**document library** Template
102	Survey Template
103	Links Template
104	Announcements Template
105	Contacts Template
106	Events Template
107	Tasks Template
108	Discussion Template
109	Image Library Template
110	Data Sources Template
111	Web Template Catalog Template
112	User Info Catalog Template
113	**Web Part** Catalog Template
114	List Template Catalog Template
115	XML **Form** Template
116	**master page** Catalog Template
117	No Code Workflows Template
118	Workflow Process Template
119	Webpage Library Template
120	Custom Grid Template
130	Data Connection Library Template

Value	Meaning
140	Workflow History Template
150	Gantt Tasks Template
200	Meetings Template
201	Agenda Template
202	Meeting User Template
207	Meeting Objective Template
210	Textbox Template
212	Homepage Library Template
1100	Issue Tracking Template
2002	My Documents Template
2003	Private Documents Template
-1	Invalid Template

Other values are not in use by the protocol. The server MUST return a value from this table to the client on request. This value MUST be set by the server when a List is created, and cannot be changed by the client.

2.2.2.3.55 vti_listtitle

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Folder

The **vti_listtitle** metakey contains the display name of the **list**.

This value MUST be used by clients to display the name of the folder in a user interface.

2.2.2.3.56 vti_longfilenames

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_longfilenames** metakey is a flag that indicates whether the server supports long file names of up to 255 characters. If the value of this metakey is 1, the server supports long file names. If the value of this metakey is 0, the server does not support long file names.

The server MUST set this value to 0 if it only supports short filenames; otherwise, it SHOULD set this value to 1.

The client MUST<10> send only short filenames when dealing with a server reporting 0 for this value.

Following is an example:

```
[vti_longfilenames;IX|1]
```

2.2.2.3.57 vti_masterurl

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	Service

The **vti_masterurl** metakey is the **server-relative URL** for a default **master page**, if any has been set. The **site** MAY specify a default master webpage to be applied within ASP.NET code pages (.aspx files) using the ~master/default.master token. This token will be expanded to the URL contained in this metakey value.

The client MAY set this value for a site. The server MUST store and return this value on request.

2.2.2.3.58 vti_metatags

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	File

The **vti_metatags** metakey is a list of the META element tag settings for the current **document**, if any.

For HTML files, the server SHOULD maintain a list of META element tags in the file. If the server maintains this list, this key MUST be present and MUST have a pair of entries for each META element tag. For META element tags that have the HTTP-EQUIV attribute, the first string in the pair MUST be "HTTP-EQUIV=" followed by the value of the HTTP-EQUIV attribute; the second string in the pair MUST be the value of the CONTENT attribute. For META element tags that have NAME and CONTENT attributes, the first string in the pair MUST be the value of the NAME attribute and the second MUST be the value of the CONTENT attribute.

A client MAY alter the way it displays files based on this value.

The server MUST parse the document for this value and MAY cache the value for return to the client on request. The client cannot set this value directly, but MAY change it by updating the document.<11>

Following is an example:

```
[vti_metatags;VR|HTTP-EQUIV=Content-Type text/html;\ charset=utf-16
HTTP-EQUIV=Content-Language en-us]
```

2.2.2.3.59 vti_modifiedby

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_modifiedby** metakey is the login name of the user who last made changes to the **page**.

The server MUST record this value as the authenticated username associated with the client whenever the client updates the **document**. The client cannot change this value directly.

2.2.2.3.60 vti_nexttolasttimemodified

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File

The **vti_nexttolasttimemodified** metakey is the next-to-last time that the **document** was changed.

The server MUST update this value with the previous time stamp for the document whenever the document is changed or updated. The client MAY use this value to determine if the document has been changed on the server by some other process since its most recent cached update.

2.2.2.3.61 vti_originator

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_originator** metakey is a string that contains the name of the application that created the original **document**, if it can be determined by parsing the HTML document for the CONTENT attribute of a META element tag with a NAME attribute of "originator" (case is not significant).

The server MUST parse the document and MAY cache this value to send on request by the client. The client cannot change this value directly, but can change it by updating the document.

Following is an example:

```
[vti_originator;SR|Microsoft Word 12]
```

2.2.2.3.62 vti_progid

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_progid** metakey is a string that contains the name of the application that created the original **document**, if it can be determined by parsing the HTML document for the CONTENT attribute of a META element tag with a NAME attribute of "progid" (case is not significant).

The server MUST parse the document and MAY cache this value to send on request by the client. The client cannot change this value directly, but can change it by updating the document.

Following is an example:

```
[vti_progid;SR|FrontPage.Editor.Document]
```

2.2.2.3.63 vti_scnoprompt

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_scnoprompt** metakey is a flag that indicates whether to prompt the user for source control input when opening a document.

A value of 0 means the client SHOULD prompt the user to check out the document when opening a document that is under source control. A value of 1 means the client SHOULD NOT prompt the user.

The server MAY set this value to recommend the client default behavior.

2.2.2.3.64 vti_servercharsets

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_servercharsets** metakey contains a list of the names of the character sets supported on the server (as specified in [RFC2616] section 3.4), separated by spaces.

The server MUST supply this list from server configuration information upon client request.

Following is an example:

```
[vti_servercharsets;VX|windows-1257 big5 windows-1252 windows-1254 iso-8859-2
iso-8859-15 windows-874 shift_jis utf-8 windows-1251 windows-1256
euc-kr gb2312 windows-1253 windows-1258 koi8-r gb18030 iso-2022-jp
ks c 5601-1987 windows-1250 windows-1255 euc-jp unicode
unicodeFFFE]
```

2.2.2.3.65 vti_serverlanguages

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)

Attribute	Value
Client Access	Read-only
Applies to	Service

The **vti_serverlanguages** metakey contains a list of supported language-codes as specified in [RFC2616] section 3.10 on the server.

The server MUST supply this list from server configuration information about client request.

Following is an example:

```
[vti_serverlanguages;VX|en-us]
```

2.2.2.3.66 vti_servertz

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_servertz** metakey contains a string with the time zone set on the server.

The server MUST supply this value from server configuration information upon client request.

Following is an example:

```
[vti_servertz;SX|-0800]
```

2.2.2.3.67 vti_setuppath

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_setuppath** metakey is used as a flag to indicate that the **document** is based on a templated document.

If the server supports templated documents, it SHOULD <12> set this value to a nonempty string for each templated document. The client MAY check that a value exists in this metakey to determine that this document is a templated document or an untemplated document. The client MAY use this value in combination with a document's **vti_hasdefaultcontent** (section 2.2.2.3.36) metakey to indicate to the user that the untemplated document MAY be reverted to the templated document state.

2.2.2.3.68 vti_sitecollectionurl

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)

Attribute	Value
Client Access	Read-only
Applies to	Service

The **vti_sitecollectionurl** metakey is the URL of the **site collection** that this **site** is a member of. This allows clients to construct and use site collection-relative URLs.

If the site is a member of a site collection, the server MUST send the **server-relative URL** to the containing site collection in this metakey to the client on request.

2.2.2.3.69 vti_sourcecontrolcheckedoutby

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_sourcecontrolcheckedoutby** metakey is the logon name of the user who opened the **page** under source control.

The server MUST include this value only if a file is checked out either short term or long term. The server MUST record the authenticated logon username of the client when a **document** is checked out and store it in this metakey for return to the client on request.

The value stored in this metakey MUST be comparable to **vti_username** (section 2.2.2.3.96). The values SHOULD be the same only if the user making the request has the file checked out.

Following is an example:

```
[vti_sourcecontrolcheckedoutby;SX|CORPDOMAIN\\johnsmith]
```

2.2.2.3.70 vti_sourcecontrolcheckincomment

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_sourcecontrolcheckincomment** metakey is a string that contains the contents of the *comment* parameter to the **checkin document** (section 3.1.5.3.6) method or **put document** (section 3.1.5.3.25) method most recently used to check in an update to this **document**.

The server MUST record this value from the *comment* parameter to the checkin document method or the put document method that updates the document. The server MUST return this value on request by the client.<13>

The client cannot set this value directly; it can be set only by using the checkin document method or the put document method.

2.2.2.3.71 vti_sourcecontrolcheckouttolocal

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	File

The **vti_sourcecontrolcheckouttolocal** metakey is a flag that indicates that the client has checked out the **document** and made a local copy to edit.

Normal checkout semantics do not indicate whether the client has a local copy of the document. This flag is set by the server to indicate that the client is editing a local copy rather than the copy of the document that the server keeps in the source control sandbox. This is also referred to as an offline long-term checkout.<14>

2.2.2.3.72 vti_sourcecontrollockexpires

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File

The **vti_sourcecontrollockexpires** metakey contains the time that the user requested for the short-term lock expiration for this file.

The server MAY use a different lock expiration time internally, to enforce minimum or maximum allowed short-term checkout periods, which means this value cannot be used to determine the remaining time available on a short-term lock.

The server MUST report the expiration time requested by the client for a short-term lock in this metakey if a short-term lock is present; the server MUST return an empty value if no short-term lock is present. The client MAY use the presence of any value in this metakey as a flag that indicates that the **document** has a current short-term lock.

2.2.2.3.73 vti_sourcecontrolmultiuserchkoutby

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	File

The **vti_sourcecontrolmultiuserchkoutby** metakey is a list of the logon usernames of the users who opened the **page** under multi-user source control.

If the server supports multi-user source control, the server MUST return a list of all usernames that have checked out this **document**. The server MUST create this list by adding the authenticated username of the client used for each checkout of the document and removing the authenticated username of the client for each checkin that does not keep the document checked out.

A client cannot set this value directly; it is set by the server when the document is checked out.

2.2.2.3.74 vti_sourcecontrolproject

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_sourcecontrolproject** metakey is a string that indicates the name of the source control project in use on the server.

The server MUST supply the default value "<STS-based locking>" on client request.

2.2.2.3.75 vti_sourcecontrolsystem

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_sourcecontrolsystem** metakey contains a string that specifies that source control is available on the server. Any nonempty string indicates to the client that source control is present on the server.

The server MUST return a nonempty string value for this metakey if source control is supported, and MUST return an empty string if source control is not supported.

2.2.2.3.76 vti_sourcecontroltimecheckedout

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File

The **vti_sourcecontroltimecheckedout** metakey is the time that the **document** was checked out on the server.

A server MUST include this value only if a file is checked out either short-term or long-term. When it is set, it MUST provide a time stamp that indicates when the file was checked out.

The client cannot set this value directly, but sets it as a side effect of a **checkout document** (section 3.1.5.3.7) method or **get document** (section 3.1.5.3.11) method with a checkout parameter set.

Following is an example:

```
[vti_sourcecontroltimecheckedout;TR|03 Nov 2007 18:10:48 -0000]
```

2.2.2.3.77 vti_themeaggregate

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)

Attribute	Value
Client Access	Read-only
Applies to	File

The **vti_themeaggregate** metakey is the **THEME-AGGREGATE-SPECIFICATION** (section 2.2.2.2.25) applied to the HTML **document**, if any.

The server MUST obtain this string by parsing the document for a META element tag with a NAME attribute of "Microsoft Theme" or "Microsoft Theme 2.00" and returning the value of the CONTENT attribute. The server MAY cache this value for return to the client on request.

This value cannot be set by the client directly but MAY be set using the **apply theme** (section 3.1.5.3.5) method.

Following is an example:

```
[vti_themeaggregate;SR|default]
```

2.2.2.3.78 vti_themedefault

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_themedefault** metakey is the **THEME-SPECIFICATION** (section 2.2.2.2.24) of the current default theme for the **site**.

The server MUST maintain this information and SHOULD return it to the client on request. The default value MUST be THEME-NONE (section 2.2.2.2.24) if no theme has been applied to the site.

This value MAY be set by the client directly, or MAY be set using the **apply theme** (section 3.1.5.3.5) method.

2.2.2.3.79 vti_thicketdir

Attribute	Value
Type	**STRING** (section 2.2.2.1.5), **BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-only
Applies to	File, **Folder**

The **vti_thicketdir** metakey contains different content depending on whether it appears in the metadata for a **document** or a folder.

Folder

When applied to a folder, this metakey contains a **BOOLEAN** flag that specifies whether the folder contains the supporting files for a **thicket**.

The server SHOULD include this metakey and set its value to TRUE for a folder that contains files that support an HTML thicket. For folders that do not contain supporting files, the server SHOULD omit this

but MAY send the key as FALSE. The client SHOULD adapt its rendering so that it does not show folders for which this key is TRUE.

Document

When applied to a document, this metakey contains a **STRING** with the name of the folder being used for supporting thicket files.

The server SHOULD include this metakey for a document that is the main file of an HTML thicket.

The server SHOULD update document and folder metakeys to indicate the presence and relationship of an HTML thicket document and its thicket supporting folder when handling the **put document** (section 3.1.5.3.25) method with a *put_option* value of "thicket".

Following is an example:

```
[vti_thicketdir;SW|jobs/index_files]
```

2.2.2.3.80 vti_thicketsupportingfile

Attribute	Value
Type	**BOOLEAN** (section 2.2.2.1.3)
Client Access	Read-Write
Applies to	File

The **vti_thicketsupportingfile** metakey indicates whether or not the **document** is a supporting file in an HTML **thicket**.

The server SHOULD include this key and set its value to TRUE for a file that is a supporting file in an HTML thicket. For files that are not supporting files, the server SHOULD omit this but MAY send the key as FALSE. The client SHOULD adapt its rendering so that it does not show files for which this key is TRUE.

The server SHOULD update document and **folder** metakeys to indicate the presence and relationship of an HTML thicket document and its thicket supporting folder when handling the **put documents** (section 3.1.5.3.26) method with a *put_option* value of "thicket".

Following is an example:

```
[vti_thicketsupportingfile;BW|false]
```

2.2.2.3.81 vti_timecreated

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File, **Folder**

The **vti_timecreated** metakey is the time that the **document** or folder was created.

The server SHOULD include this metakey for files and folders. It SHOULD reflect the time that the file or folder was created. The client MAY use this value to render details about files or folders.

Following is an example:

```
[vti_timecreated;TR|24 Apr 2000 15:54:33 -0000]
```

2.2.2.3.82 vti_timelastmodified

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File, **Folder**

The **vti_timelastmodified** metakey is the time that the **document** or folder was most recently modified.

The server SHOULD include this key for files and folders. It SHOULD approximate the date and time that the file or folder was last modified.

As specified in section 2.2.2.2.18, under the edit and overwrite values, the server MUST use this value for concurrency control and thus might not be able to make it reflect the time that the file was last modified. The client MAY use this value to render details about files or folders; however, **vti_timelastwritten** (section 2.2.2.3.83) is often more appropriate when it is available. The client SHOULD send this value in accordance with its use, as specified in section 2.2.2.2.18, under the edit and overwrite values.

Following is an example:

```
[vti_timelastmodified;TR|24 Apr 2000 15:54:33 -0000]
```

2.2.2.3.83 vti_timelastwritten

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	File, **Folder**

The **vti_timelastwritten** metakey is the time that the **document** or folder was last saved to local storage on the server.

The server SHOULD include this metakey for files and folders. The server SHOULD record the date and time that the file or folder content was modified in this metakey. The client SHOULD use this value to render details about files or folders but MAY use **vti_timelastmodified** (section 2.2.2.3.82) for that purpose.

Following is an example:

```
[vti_timelastwritten;TR|24 Apr 2000 15:54:33 -0000]
```

2.2.2.3.84 vti_title

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-write
Applies to	File, Service

The **vti_title** metakey is the user-readable title of a **document** or **site**.

The server SHOULD maintain this key for a site as a user-readable description of the site. The client MAY use this string to refer to the site when presenting information to a user.

The server SHOULD maintain this key for documents. If the document is an HTML document on the server, the server SHOULD parse the document for the content of a TITLE element tag, and MAY cache this value for return to the client. The client MAY update this metakey for HTML documents and the server SHOULD rewrite the document with an updated TITLE element.

For file streams that the server is unable to parse, the server SHOULD accept a client-supplied metakey value and return it on client request.

The client SHOULD use this metakey value where the title of the document is to be displayed to the user.

Following is an example:

```
[vti_title;SR|Trey Research]
```

2.2.2.3.85 vti_toolpaneurl

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_toolpaneurl** metakey is the URL of the site's Web Parts Tool Pane page.

If the server supports **Web Parts**, the server MUST return the URL of the Web Parts Tool Pane to the client on request. The client MAY make user interface options available for browsing to this page if this metakey has a value. <15>

2.2.2.3.86 vti_usagebyday

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagebyday** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of **page** views for this **site** for up to 32 days, beginning with the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

Following is an example:

```
[vti_usagebyday;UX|0 0 0 0 6 2]
```

2.2.2.3.87 vti_usagebymonth

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagebymonth** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of **page** views for this **site** for up to 32 months, beginning with the month containing the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

Following is an example:

```
[vti_usagebymonth;UX|1 0 3 6 4 5 10 2]
```

2.2.2.3.88 vti_usagedownload

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagedownload** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of kilobytes downloaded from this **site** by month for up to 32 months, beginning with the month containing the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

2.2.2.3.89 vti_usagefirstdatadaycount

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_usagefirstdatadaycount** metakey contains an integer that represents the daycount number for the first (that is, most recent) day represented in the usage data.

The server MUST set this metakey value to the daycount of the most recent completed day when usage stats are updated.

2.2.2.3.90 vti_usagehitsbyday

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagehitsbyday** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of hits for this site for up to 32 days, beginning with the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

2.2.2.3.91 vti_usagehitsbymonth

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagehitsbymonth** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of hits for this **site** for up to 32 months, beginning with the month containing the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

2.2.2.3.92 vti_usagelastupdatedaycount

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

The **vti_usagelastupdatedaycount** metakey is the daycount of the most recent completed day for which usage stats have been updated.

The server MUST set this metakey value to the daycount of the most recent completed day when usage stats are updated.

2.2.2.3.93 vti_usagelastupdatedonet

Attribute	Value
Type	**TIME** (section 2.2.2.1.6)
Client Access	Read-only
Applies to	Service

The **vti_usagelastupdatedonet** metakey contains the time the server last updated the usage statistics.

The server MUST set this metakey value when usage stats are updated.

2.2.2.3.94 vti_usagevisitsbyday

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagevisitsbyday** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of visits to this site for up to 32 days, beginning with the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

2.2.2.3.95 vti_usagevisitsbymonth

Attribute	Value
Type	**METADICT-INT-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_usagevisitsbymonth** metakey contains an array of **UNSIGNED-INT** (section 2.2.2.1.1) that represents the number of visits to this **site** for up to 32 months, beginning with the month containing the day represented by the daycount contained in the **vti_usagefirstdatadaycount** (section 2.2.2.3.89) metakey, and going backward.

If the server is configured to provide usage statistics, it SHOULD update this value once a day.

2.2.2.3.96 vti_username

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	Service

The **vti_username** metakey is the current authenticated logon username associated with the client.

The server SHOULD include this metakey in the **site** metadata. When the key is present, the client SHOULD use this key for comparisons with **vti_sourcecontrolcheckedoutby** (section 2.2.2.3.69).

The server SHOULD use and maintain its own value rather than use a client-supplied value for this key.

Following is an example:

```
[vti_username;SX|CORPDOMAIN\\johnsmith]
```

2.2.2.3.97 vti_usernames

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	File

The **vti_usernames** metakey contains a list of all the usernames that have ever been associated with this **document** by the **vti_assignedto** (section 2.2.2.3.11) metakey.

The server SHOULD insert the username found in the **vti_assignedto** metakey whenever it is updated on the document into the array of values stored for the **vti_usernames** metakey.

Following is an example:

```
[vti_usernames;VX|CORPDOMAIN\\smith CORPDOMAIN\\jones]
```

2.2.2.3.98 vti_virusinfo

Attribute	Value
Type	**STRING** (section 2.2.2.1.5)
Client Access	Read-only
Applies to	File

The **vti_virusinfo** metakey contains a string with information about the virus infection status of this **document**.

This string is provided by the virus-checking service running on the server, if any. The server SHOULD pass on the descriptive information provided by the virus-checking service in this metakey.

2.2.2.3.99 vti_virusstatus

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	File

The **vti_virusstatus** metakey contains an integer that represents the status of the file as determined by the virus-checking service on the server, if any, according to the following table.

Value	Meaning
0	The file is clean.
1	The file is infected.
2	The file is infected, but cleanable.
3	The file has had a virus infection cleaned.
4	The attempt to clean the virus from this file has failed.

Value	Meaning
5	The file has been deleted by the virus check.
6	The virus check has timed out.

The server SHOULD translate the status information provided by the virus-checking service into one of these codes and store it in this metakey.

2.2.2.3.100 vti_welcomenames

Attribute	Value
Type	**METADICT-STRING-VECTOR** (section 2.2.2.2.11)
Client Access	Read-only
Applies to	Service

The **vti_welcomenames** metakey is the name or names of the default web page or pages used by the **site**.

All sites have a default page that displays when first accessed. This default page (often called the home page) is usually index.htm or default.htm, although it can have any name. Web servers, in general, enable using a list of file names that can be default pages.

The server MUST provide the list of default web page names to the client on request.

Following is an example:

```
[vti_welcomenames;VX|default.htm default.aspx]
```

2.2.2.3.101 mf-file-status

Attribute	Value
Type	**INT** (section 2.2.2.1.2)
Client Access	Read-only
Applies to	Service

Description:

The current file transfer status, sent as part of the **get documents** (section 3.1.5.3.12) method server response. A value of 0 indicates success. Any nonzero value indicates a failure.

2.2.2.4 Irrecoverable Error Responses

When a request is not syntactically valid (for example, if a string other than "true" or "false" is given when a **BOOLEAN** (section 2.2.2.1.3) value is expected), when the server is required to allocate more memory than it can, or when the server finds itself in some other irrecoverable failure situation, the server MUST abandon further processing of the client request. If the server encounters an error and cannot proceed with the request, it MUST return a STATUS (section 2.2.2.2.17) in HTML Mode (section 2.2.1.1.2). The client SHOULD recognize a STATUS returned by the server as an indication that the server processing of the request failed in some way and SHOULD abandon any parsing context it was in.

3 Protocol Details

The following sections describe several operations as used in common scenarios to illustrate the function of the FrontPage Server Extensions: Website Management Protocol.

3.1 Common Details

A FrontPage Server Extensions: Website Management Protocol client SHOULD initialize a connection with the server. The client MAY then make as many method calls against the server as needed. The FrontPage Server Extensions: Website Management Protocol, like HTTP 1.1 (as specified in [RFC2616]), is a stateless protocol. As such, connections do not need to be closed.

3.1.1 Abstract Data Model

The FrontPage Server Extensions: Website Management Protocol has three navigational concepts in its site administration system: files, directories, and services. Each of these structures MUST have a metadictionary of metadata associated with it. This metadata can be used by clients or servers to store whatever information is relevant to the object.

The metadictionary is also often used as a way for the server to communicate information about a file to the client, such as its checkout state.

In addition to the metadictionary, the following are the relevant properties of each file system concept:

- Files MUST have a content stream, that is, an array of sequenced bytes that represents the contents of the file.
- Directories MUST NOT have a content stream, but MUST be able to contain files or other directories.
- Services model the concept of a site. Like directories, they MUST NOT have a content stream and MUST be able to contain files or directories. Services MUST also be able to answer questions about their own capabilities, for example, if they support source control. The capabilities of a service are communicated to clients by using the metadictionary of the service.

3.1.1.1 Source Control

Servers MUST support short-term checkout and SHOULD implement a source control sandbox. The server MAY offer users the option to turn off the source control sandbox.<16>

For a server that has the source control sandbox turned off:

When a **document** is checked out, the server MUST NOT comply with any operations on the document that are not sent by the user who has the document checked out, including another user requesting to read the document.

For a server that has the source control sandbox turned on:

When a document is checked out, the server MUST NOT comply with any requests to modify the document that are not sent by the user who has the document checked out. However, if another user merely requests the contents of the document, the server SHOULD respond with the document stream as it appeared when the document was checked out.

A document that is checked out short-term can have the checkout released in either of two ways: The client can send an **uncheckout document** (section 3.1.5.3.40) request to the server, or the short-term checkout expires.

3.1.2 Timers

3.1.2.1 Short-Term Checkout Timer

A **document** checked out short-term has a time-out value associated with the checkout. The length of the short-term checkout timer is determined based on client input, as specified in section 3.1.5.3.7.<17>

3.1.3 Initialization

Microsoft FrontPage Server Extensions initialization requires the following operations to take place.

3.1.3.1 Determining Server Capabilities

A prospective client MAY perform an HTTP OPTIONS request, as specified in [RFC2616] section 9.2, against a server to determine if it is a FrontPage Server Extensions: Website Management Protocol server.

When receiving an OPTIONS request, a FrontPage Server Extensions: Website Management Protocol server MUST return the header "MS-Author-Via:" with a value that includes "MS-FP/4.0" to indicate that the server supports the FrontPage Server Extensions: Website Management Protocol, as specified in [MS-WDVSE] section 3.2.5.2. The server SHOULD list the protocols in the MS-Author-Via header in order of preference from highest to lowest. The client SHOULD use the first client-supported protocol listed in the MS-Author-Via header. Results of the HTTP OPTIONS request that are returned from the server SHOULD be cached by clients as a performance optimization.

3.1.3.2 Determining Entry Points

Each method call is an HTTP POST by the client to a URL on the server. On any server there are four entry points, which can be discovered by the clients (as specified in section 4.1.1). Each method entry denotes which entry point the protocol requires to be used to call that method. The four entry points are detailed in the following table.

Name	Description
FPShtmlScriptUrl	Used to retrieve the **server version** (section 3.1.5.3.36) method; also for the **url to web url** (section 3.1.5.3.41) method.
FPAuthorScriptUrl	Used for all methods to deal with **document** manipulation.
FPAdminScriptUrl	Used for all methods that deal with **site** administration.
TPScriptUrl	Not used by the FrontPage Server Extensions: Website Management Protocol. See section 4.1.2.

3.1.3.2.1 Client Request for Entry Point HTML Page

If the server supports the FrontPage Server Extensions: Website Management Protocol, the client SHOULD perform an HTTP GET of vti_inf.html at the server root site by using the following to determine the entry point for the FrontPage Server Extensions.<18>

```
http://fpseserver/_vti_inf.html
```

where **fpseserver/** is the root site of the server.

3.1.3.2.2 Server Entry Point HTML Page Response

The server MUST reply to the HTTP GET of vti_inf.html with an HTML **page** that contains an HTML comment that is an ENTRY-POINT-COMMENT, as follows:

```
ENTRY-POINT-COMMENT = COMMENT-BEGIN + SHTML-ENTRY-POINT +
AUTHOR-ENTRY-POINT + ADMIN-ENTRY-POINT + TPSCRIPT-ENTRY-POINT +
COMMENT-CLOSE

COMMENT-BEGIN = "<!-- FrontPage Configuration Information FPVersion="
+ DQUOTE + VERSION + DQUOTE + LF

SHTML-ENTRY-POINT = "FPShtmlScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

AUTHOR-ENTRY-POINT = "FPAuthorScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

ADMIN-ENTRY-POINT = "FPAdminScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

TPSCRIPT-ENTRY-POINT = "TPScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF
```

Servers SHOULD return a comment that defines the entry points as follows, because clients MAY assume these values:

```
<!-- FrontPage Configuration Information FPVersion="12.0.0.000"
FPShtmlScriptUrl=" vti bin/shtml.dll/ vti rpc"
FPAuthorScriptUrl="_vti_bin/_vti_aut/author.dll"
FPAdminScriptUrl=" vti bin/ vti adm/admin.dll"
TPScriptUrl="_vti_bin/owssvr.dll" -->
```

Each method description contains a section that defines which entry point that method uses. Clients MUST post to the correct entry point, or the server SHOULD ignore their request.

The client SHOULD then call the **server version** (section 3.1.5.3.36) method on the root of the server to determine the latest server version of the protocol that the server supports. The client SHOULD use its own version number for the PROTOCOL-VERSION-STRING (section 2.2.2.2.2) of the **METHOD-KEY-VALUE** (section 2.2.2.2.6), unless and until it knows the server version number, in which case it SHOULD use whichever of the two version numbers is lower. The server MUST respond with either the same version number it received, or with its own version number, whichever is lower. Thereafter, the client SHOULD send the version number it got back from the server.

If the client is opening a site that is not at the root of the server, the client MUST call the **url to web url** (section 3.1.5.3.41) request. This method accepts a **server-relative URL**, such as "/subsite/folder/document.txt". It would return the server-relative URL of the site, "/subsite", and the service-relative URL of the item, "folder/document.txt".

The client MUST then post all further methods to the site that it wants to communicate with. The *service_name* parameter MUST NOT be used by the client to denote what site that it is communicating with, because this parameter is ignored by the server. For example, if the client wants to check out "/subsite/folder/document.txt", it needs to post to the SHTML-ENTRY-POINT of the **subsites**. Assuming the default value of the SHTML-ENTRY-POINT, that would be the following:

```
http://fpseserver/subsite/_vti_bin/shtml.dll/_vti_rpc
```

The client then refers to the file that it wants to check out by its service-relative URL within the FrontPage Server Extensions: Website Management Protocol request. Finally, the client SHOULD call the **open service** (section 3.1.5.3.24) method on the appropriate site to begin the conversation. The client MAY then make whatever method calls that it needs against the server.

3.1.4 Higher-Layer Triggered Events

There are no higher-layer triggered events for the FrontPage Server Extensions: Website Management Protocol server. Each client request is triggered by the client application's needs.

3.1.5 Message Processing Events and Sequencing Rules

Aside from initialization, methods MAY be called in any order, as determined by the client application's needs. The only methods forming a strong logical pair are the **checkout document** (section 3.1.5.3.7) and **uncheckout document** (section 3.1.5.3.40) methods. Clients that call the former SHOULD call the latter because a **document** that is checked out cannot be edited by any other users until the checkout is revoked.

Section 4 provides examples that show common operations being performed against the server, and that section also provides some guidance about common method sequences.

3.1.5.1 HTTP Headers

The client SHOULD send an X-Vermeer-Content-Type header (as specified in [RFC2616] section 14.17) with the same value as the standard HTTP Content-Type header to safeguard against one-click attacks, as specified in section 5.1. The server MUST use this header, if present, to determine the Content-Type of the request. If this header is not present, the server SHOULD fail the request.

The client MUST also include the string "FrontPage" (case-sensitive) in its User-Agent header, as specified in [RFC2616] section 14.43. The server MAY alter its responses when the client does not do this.<19>

Except as specified in specific methods, server responses MUST have the HTTP Content-Type "application/x-vermeer-rpc".

3.1.5.2 Method Formatting

A set of formatting conventions is used for each FrontPage Server Extensions method, as specified in section 3.1.5.3.

Each method begins with a brief description of the method's purpose. A Parameters section follows the description and corresponds to the **REQUEST** (section 2.2.2.2.7) element that specifies the set of arguments for each method. Clients MAY pass in any subset of the given arguments, although in some cases specific arguments are required. If an argument is not required and unless otherwise specified, the server SHOULD use a default value of FALSE for **BOOLEAN** (section 2.2.2.1.3) arguments, 0 for **INT** (section 2.2.2.1.2), **UNSIGNED-INT** (section 2.2.2.1.1), and **DOUBLE** (section 2.2.2.1.4) arguments, an empty string for **STRING** (section 2.2.2.1.5) and **URL-STRING** (section 2.2.2.2.3) arguments, an empty **METADICT (section 2.2.2.2.11)** for **METADICT** arguments, and an empty **VECTOR-X** (section 2.2.2.2.1) for **VECTOR-X** type arguments. Clients MAY pass the arguments in any order, and servers MUST accept them in any order. Clients SHOULD NOT pass arguments unless they are mentioned in this document. When a server is passed an argument that it does not recognize, the server SHOULD treat it as a syntax error. It MAY choose to ignore the parameter instead.<20>

Each argument definition starts with an argument name in italics. The argument name corresponds to the ARG-NAME element. The data type is listed in the argument description, and it defines the required format for the ARG-VALUE element.

The Entry Point section defines the URL as provided by _vti_inf.html, to which clients MUST post when using this method.

The **Return Values** section specifies the set of return values, and its formatting is similar to the Parameters section. The **RET-NAME** (section 2.2.2.2.8) corresponds to the return value name in bold. The format of the RET-VALUE is defined by the return value type. Servers MAY return arguments in any order, and clients MUST accept the parameters in any order.

In error conditions at the FrontPage Server Extensions: Website Management Protocol level, the server SHOULD return a RET-NAME RET-VALUE pair where the RET-NAME is "status" and the RET-VALUE is a **STATUS** object (as specified in section 2.2.2.2.17). A client SHOULD ignore all other return values if a status is present. Even though this is an error, it SHOULD be encapsulated in an HTTP 200 response, as specified in [RFC2616] section 10.4.2.

Lower layers (such as HTTP or IP) can also return errors. For example, the FrontPage Server Extensions: Website Management Protocol layer on the server SHOULD indicate to the HTTP layer that it requires authentication, which in turn SHOULD cause the HTTP layer to send a 401 message in response (as specified in [RFC2616] section 10.4.2) to unauthenticated requests.

If the lower layers pass on an unauthenticated request from the client to the FrontPage Server Extensions: Website Management Protocol server, it SHOULD respond with an HTTP 401, as specified in [RFC2616] section 10.4.2. It MAY include an entity body that contains a descriptive error message in the response.<21>

3.1.5.3 Methods

Each request by a client that uses the FrontPage Server Extensions: Website Management Protocol MUST begin with a field that contains the method name. For example, the **get document** (section 3.1.5.3.11) request has the string "get document" in the method field. Following the method name field is the list of arguments that MAY be specified in any order.<22>

The client POSTs the following FrontPage Server Extensions: Website Management Protocol requests to the server. These requests are specified in the following sections.

3.1.5.3.1 Common Method Parameters and Return Values

The following parameters and return values are common to many of the methods.

Parameters

apply_opt: An **APPLY-OPTION** (section 2.2.2.2.36) value that consists of one or more comma-delimited arguments that tell the server to apply a theme, style sheet, or border to an entire **site** or to one or more **documents**. An optional argument can request metadata on the object to which the theme, style sheet, or border was applied. This parameter is not required and defaults to no option.

The arguments for this parameter are as follows.

Argument	Action
APPLY-OPT-WEB	Apply the method to the entire site.
APPLY-OPT-PAGE	Apply the method to one or more pages.
APPLY-OPT-RFI	Return meta information for the affected documents.

The APPLY-OPT-WEB and APPLY-OPT-PAGE arguments are effectively mutually exclusive; if the APPLY-OPT-WEB argument is present, any APPLY-OPT-PAGE argument is ignored. If the APPLY-OPT-RFI argument is included, the return value includes the updated metadata for each page that is specified by the *url_list* parameter.

document_name: A **URL-STRING** (section 2.2.2.2.3) that specifies the service-relative URL of the document that the request is addressed to.

effective_protocol_version: This parameter is deprecated. Clients conforming to the FrontPage Server Extensions: Website Management Protocol MUST NOT send this parameter. If this parameter is sent by a client, the server SHOULD validate it as a **VERSION** (section 2.2.2.2.9) but MUST otherwise ignore it. <23>

listLinkInfo: A **BOOLEAN** (section 2.2.2.1.3) value that specifies whether the server response SHOULD contain information about the links from the current page or pages. If TRUE, the response SHOULD include link information; if FALSE, link information SHOULD be excluded to reduce bandwidth and server processing overhead. Servers MAY ignore this parameter and avoid sending link information for nonconforming clients.<24>

meta_info: A **METADICT** (section 2.2.2.2.11) that contains all the metadata that is required for the request. Details about the specific metadata returned are provided in the descriptive text for that request.

return_stats: The client MUST NOT send this **BOOLEAN** value. If the server receives it, the server MUST validate its data type but SHOULD otherwise ignore it. If a server chooses not to ignore it, it MAY return an additional return value that provides details about the cost of the request.<25>

service_name: This parameter is obsolete for all methods except **create service** (section 3.1.5.3.8), **remove service** (section 3.1.5.3.32), and **rename service** (section 3.1.5.3.33), but the client MAY send this **URL-STRING** that defines the **server-relative URL** of the site that the request is addressed to. <26>For all methods other than the ones specified, the server MUST ignore it. The URL to which the request is posted MUST be used to determine what site the request is issued against.

url_list: A **VECTOR-URL-STRING** (section 2.2.2.2.3) list of service-relative URLs that specifies which documents will have the method applied. The client SHOULD omit this parameter or leave it empty to obtain the default server behavior.

validateWelcomeNames: A **BOOLEAN** value that determines if each item in the file's list of links SHOULD replace links to a **folder** with links to the welcome page (the default web page) for that folder, if it exists. If TRUE, checking for default pages is done; if FALSE, checking is not done. The server MAY ignore this value.

Return Values

The following are common return values for several methods:

document_list: A **DOCUMENT-LIST-RETURN-TYPE** (section 2.2.2.2.13) that contains the names and metadata for all the documents and folders specified in the request. Depending on the method, these documents and folders MAY be specified as parameters, or if unspecified, MAY include all the documents and folders in a specified site.

message: A **STRING** (section 2.2.2.1.5) description of the action taken by the server. This is intended for debugging, and SHOULD be ignored by the client.

meta_info: A **METADICT** that contains a list of the metadata returned in response to the request. Depending on the method, this **METADICT** can contain all or only a specified subset of the metadata available for the target of the request, which can be a site, folder, or document, as specified in each method description.

status: A **STATUS** (section 2.2.2.2.17) that contains a description of the error result, if any. This return value SHOULD be included only for any method that encounters an error during processing. The client MAY display an error message to the user, log the error, or attempt to recover from the error.

3.1.5.3.2 add document to source control

Adds a document to a source control database. This method is deprecated, because all documents are under source control.<27>

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

document_name: For semantics, see *document_name* in section 3.1.5.3.1. The client MUST send this parameter, and the server SHOULD interpret this **URL-STRING** (section 2.2.2.2.3) as the absolute URL or service-relative URL of the document to add to source control.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** (section 2.2.2.2.11) contains all the metadata available to the client for the document after it has been added to source control.

3.1.5.3.3 apply border

Designates the top, bottom, left, or right side of a **page** or pages as reserved. Regular margins are adjusted so the designated border space is not used for normal page development.<28>

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

border_spec: A **BORDER-SPECIFICATION** (section 2.2.2.2.21) value that identifies the borders in use on a page.

url_list: For semantics, see *url_list* in section 3.1.5.3.1. This parameter specifies the **documents** to which the server MUST apply the *border_spec* argument if the *apply_opt* argument does not include APPLY-OPT-WEB.

If this parameter is omitted or left empty, the border-specification MUST be set on the server as the default and applied to all of the applicable documents on the **site**.

apply_opt: For semantics, see *apply_opt* in section 3.1.5.3.1. If the *apply_opt* argument includes APPLY-OPT-WEB, the *border_spec* argument is applied to all of the documents on the site, regardless of the value of the *url_list* parameter. If the *apply_opt* argument does not include APPLY-OPT-WEB, the server MUST apply the *border_spec* argument according to the argument specified for the *url_list* parameter.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. This value is returned only if the *apply_opt* arguments include APPLY-OPT-RFI and the *url_list* argument is not empty. The **document_list** contents are specified by the *url_list* argument.

3.1.5.3.4 apply stylesheet

The apply stylesheet request is used by the client to apply a set of **CSS** to an entire **site** or to specified **documents**.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

link_list: A **VECTOR-URL-STRING** (section 2.2.2.2.3) that lists the URLs in absolute URL format, server-relative format, or service-relative format for the CSS to be applied to all the documents in the entire site, or to the documents specified by the *url_list* argument, depending on the value of the *apply_opt* argument.

url_list: For semantics, see *url_list* in section 3.1.5.3.1. Specifies the documents to which the server MUST apply the cascading style sheets specified in the *link_list* argument if the *apply_opt* argument does not include APPLY-OPT-WEB.

If this parameter is omitted or left empty, the style sheets specified by the *link_list* argument MUST be applied to all of the applicable documents on the site.

apply_opt: For semantics, see *apply_opt* in section 3.1.5.3.1. If the *apply_opt* arguments include APPLY-OPT-WEB, the style sheets specified by the *link_list* arguments will be applied to all of the documents on the site, regardless of the *url_list* arguments. If the *apply_opt* arguments do not include APPLY-OPT-WEB, the server MUST apply the style sheets according to the arguments specified for the *url_list* parameter.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

link_list: A **VECTOR-URL-STRING** that contains the argument to the *link_list* parameter.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. This value is returned only if the *apply_opt* argument includes APPLY-OPT-RFI and the *url_list* argument is not empty. APPLY-OPT-RFI MUST NOT be specified unless the *url_list* is non-empty. The **document_list** contents are specified by the *url_list* argument.

3.1.5.3.5 apply theme

This method specifies which theme to apply to the **site** or the specified **documents**. The client MAY also send theme-parameters information about how the theme is to be implemented, including whether or not to use **CSS** to create the theme.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

theme_name: A **STRING** (section 2.2.2.1.5) containing the **THEME-SPECIFIER** (section 2.2.2.2.24) of the theme to be applied to the documents specified by the *url_list* and *apply_opts* arguments.

theme_params: A **THEME-PARAMETERS** (section 2.2.2.2.23) that contains the client specification for applying themes that use CSS, color type, active graphics, and background type. If this argument is omitted, the server MUST default to a **THEME-PARAMETERS** value of 1000. This argument MUST be ignored by the server if the theme_name argument is THEME-NONE (section 2.2.2.2.24).

url_list: For semantics, see section 3.1.5.3.1. Specifies the documents to which the server MUST apply the theme specified by the theme_name argument, if the *apply_opt* argument does not include APPLY-OPT-WEB.

If this argument is omitted or left empty, the theme MUST be set on the site as the default, and MUST be applied to all of the documents on the site.

apply_opt: For semantics, see section 3.1.5.3.1. If the *apply_opt* argument includes APPLY-OPT-WEB, then the theme will be applied to all of the documents on the site, regardless of the value of the *url_list* argument. If the *apply_opt* argument does not include APPLY-OPT-WEB, then the server MUST apply the theme according to the argument specified for the *url_list* parameter.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

office_themed_documents: A **VECTOR-URL-STRING** (section 2.2.2.2.3) containing the service-relative URLs of Microsoft Office documents that did not have the theme applied, because they MUST be themed with a Microsoft Office application.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. This value is returned only if the *apply_opt* arguments include APPLY-OPT-RFI and the *url_list* argument is not empty. This return value contains the names and metadata for all the documents which the theme has been applied to if they were specified with the *url_list* argument. If the **document_list** is empty, the theme has not been applied to any documents.

3.1.5.3.6 checkin document

The checkin document request is used by the client to enable the currently authenticated user to check in and unlock a document under source control which was previously checked out for editing using the **checkout document** (section 3.1.5.3.7) request or the **get document** (section 3.1.5.3.11) request with a "checkout" argument in the *get_option* parameter. The server MUST NOT check in a file if it has a short term lock.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

document_name: For semantics, see *document_name* in section 3.1.5.3.1.The client MUST send and the server MUST interpret this **URL-STRING** (section 2.2.2.2.3) as the service-relative path of the document to check in.

comment: A **STRING** (section 2.2.2.1.5) that provides a checkin comment for the file being checked in. This parameter MAY be omitted by the client and defaults to an empty string.

keep_checked_out: A **BOOLEAN** (section 2.2.2.1.3) value that SHOULD determine a specified document's behavior in source control. If TRUE, the document SHOULD be checked in to source control and immediately checked back out; if FALSE, the document SHOULD be checked in. The server MUST treat this as equivalent to the "checkout" **PUT-OPTION-VAL**, as specified in **Put-Option** (section 2.2.2.2.18). Clients MAY omit this parameter, which defaults to FALSE.

time_checked_out: A **TIME** (section 2.2.2.1.6) that indicates the client's record of the time and date at which the file was last checked out. The server MAY refuse to commit the checkin if the time does not match the server's record of the time the file was checked out.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** (section 2.2.2.2.11) contains all the metadata available to the client for the document that has been checked in after the operation has been completed.

3.1.5.3.7 checkout document

The **checkout document** request is used by the client to enable the currently authenticated user to lock a **document** and prevent other users from making changes to the document while it is locked.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

document_name: For semantics, see *document_name* in 3.1.5.3.1. The client MUST send and the server MUST interpret this **URL-STRING** (section 2.2.2.2.3) as the service-relative path of the document to checkout.

force: This parameter is an **INT** (section 2.2.2.1.2) bit mask; bits 0 and 1 are defined as follows.

Mask	Meaning
Bit 0 0x00000001	Force checkout. This feature is currently unimplemented and reserved for future use. Clients MUST NOT set this bit, and servers MUST ignore it.
Bit 1 0x00000002	Refresh short-term checkout by the currently authenticated user. The client MUST set this bit if, and only if, it already has a short-term checkout on the file and there is a requirement to extend the time-out on it. The server MUST attempt to create a new short-term checkout if this bit is cleared, and it MUST attempt to extend an existing short-term checkout if this bit is set. It MUST return an error if this bit is set when no short-term checkout exists; it also MUST return an error if this bit is not set and a short-term checkout does exist.

The client MUST set all unused bits to 0, and the server MUST ignore all unused bits. Range: from -2147483648 through 2147483647; only 0 and 2 are currently allowed.

timeout: An INT that defines the number of minutes that a short-term checkout is requested. To retain the lock, the client MUST renew its short-term checkout within this interval. The client MUST NOT send negative values, and the server MUST ignore them. Servers MAY ignore requests in which this parameter is set to 0, which represents a request for a long-term checkout. Range: 0 to 2147483647.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** (section 2.2.2.2.11) contains all the metadata available to the client for the document that has been checked out, after the operation has completed.

3.1.5.3.8 create service

The **create service** method is used to create a new **subsite** on the server.

Parameters

service_name: For semantics, see *service_name* in section 3.1.5.3.1. The client SHOULD send this parameter as a suggestion for the name of the new subsite if the *flags* parameter is sent with a value of 1. The client MUST NOT send this parameter if the *flags* parameter is sent with a value of 0.

meta_info: For semantics, see *meta_info* in section 3.1.5.3.1. This **METADICT** (section 2.2.2.2.11) contains the metadata that the server MUST apply to the newly created service subsite. If this parameter is missing or empty, the server MUST create the service subsite with all default values.

flags: Contains an **INT** (section 2.2.2.1.2) used by the client to indicate whether the server can treat the name provided in the *service_name* parameter as a suggestion for the name of the newly created subsite. If the flag has a value of 1, and if the value of the *service_name* parameter contains unacceptable characters, the server SHOULD use a similar name to create the subsite. If this flag is 0, and if the *service_name* parameter is not valid, the server MUST NOT create the subsite.

lcid: An **INT** (section 2.2.2.1.2) that contains the **LCID** that the server SHOULD use as the default for the new subsite. By default, the server SHOULD use the LCID value of the parent **site**.

Entry Point

FPAdminScriptUrl

Return Values

service_name: A **URL-STRING** (section 2.2.2.2.3) that contains the **server-relative URL** for the newly created subsite.

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** contains all the newly created subsite's metadata available to the client.

3.1.5.3.9 create url-directories

The **create url-directories** request allows the client to create one or more directories (**folders**) on the **site** along with the specified metadata.

This operation is not atomic. If the bulk operation fails, some folders might have been created. In the case of a failure, the client SHOULD query the server with **list documents** (section 3.1.5.3.20) if it needs to determine what folders were created. Clients SHOULD use this method rather than **create url-directory** (section 3.1.5.3.10), to minimize the number of remote procedure calls (RPCs) made to create folders and to set their metadata.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

urldirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) specifying the names and metadata of folders to create. The client MUST specify one **URL-DIRECTORY** for each folder it wants to create. The server SHOULD create the folders in the order specified. The client MUST send this parameter.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

3.1.5.3.10 create url-directory

The create url-directory request is used by the client to create a **folder** for the current **site**. Clients SHOULD use create url-directories (section 3.1.5.3.9) instead of this method to take advantage of the ability to set metadata for the folder without requiring a second RPC method call. However, servers MUST support this method for backward compatibility.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

url: A **URL-STRING** (section 2.2.2.2.3) that specifies the URL of the folder to be created. The client MUST send this parameter.

executable: A **BOOLEAN** (section 2.2.2.1.3) that indicates if the security settings for the newly created folder SHOULD allow execution of entities within it. If TRUE, the request is to create an executable folder. Clients that conform to the FrontPage Server Extensions: Website Management Protocol MUST send FALSE. Servers MUST ignore this for architectures that do not support the notion of an executable folder. The server SHOULD<29> ignore this for security reasons.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

urldir: A URL-DIRECTORY (section 2.2.2.2.16) that contains the metadata for the folder that was created.

3.1.5.3.11 get document

The **get document** request is used by a client to retrieve a **document** and its metadata for viewing or editing on the client. The document stream is sent after the </HTML> tag at the end of the standard response.

Parameters

service_name: This parameter is deprecated. For semantics, see *service_name* in section 3.1.5.3.1.

document_name: For semantics, see *document_name* in section 3.1.5.3.1. This is the service-relative URL of the document to retrieve.

effective_protocol_version: This parameter is deprecated. For semantics, see *effective_protocol_version* in section 3.1.5.3.1.

old_theme_html: This parameter is no longer used by the protocol. The server MUST validate any argument sent as a **BOOLEAN** (section 2.2.2.1.3) but MUST ignore the result.

expandWebPartPages: A **BOOLEAN** value reserved for future use. This parameter MUST be ignored by the server regardless of a TRUE or FALSE value. The client MUST send FALSE for this parameter, either explicitly or by omitting the parameter.

force: A **BOOLEAN** value reserved for future use. This parameter MUST be ignored by the server, regardless of a TRUE or FALSE value. The client SHOULD either send FALSE for this parameter or omit the parameter.

doc_version: A **STRING** (section 2.2.2.1.5) that contains a **SOURCE-CONTROL-DOCUMENT-VERSION** (section 2.2.2.2.27) for the document being retrieved. An empty string (the default value)

is a request for the current version of the document. A server SHOULD ignore this parameter and send the most recent version of the document.

get_option: A **STRING** (section 2.2.2.1.5) value that determines how documents are checked out of source control. Passing any string not listed in the following table is considered the same as "none".

Value	Meaning
none	Do not check out the file.
chkoutExclusive	Check out the file exclusively. The server MUST return an error if this flag is passed and the file is already checked out by another user.
chkoutNonExclusive	Check out the file nonexclusively, if the source control system allows non-exclusive checkouts. If it does not, the server MUST<30> treat this as chkoutExclusive instead.

The checkout MUST logically occur before the server begins sending the document to the client. If the checkout cannot occur, the server MUST send a failure and not return the document.

timeout: An **UNSIGNED-INT** (section 2.2.2.1.1) that specifies the number of minutes the server is required to retain the short-term checkout.

To retain the lock longer, the client MUST renew its short-term checkout within this interval. For details, see **checkout document** (section 3.1.5.3.7). The client MUST NOT send negative values, and the server MUST ignore them. Servers MAY ignore requests in which this parameter is set to 0, which indicates a long-term checkout. Range from 0 through 2147483647.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

document: The **DOCINFO** (section 2.2.2.2.12) that contains the document name and metadata for the document that has been retrieved.

3.1.5.3.12 get documents

The **get documents** request is used by the client to retrieve a set of **documents** for viewing on a client computer.

Parameters

service_name: This parameter is deprecated; see *service_name* in section 3.1.5.3.1.

url_list: For semantics, see *url_list* in section 3.1.5.3.1.

effective_protocol_version: This parameter is deprecated. For semantics, see *effective_protocol_version* in section 3.1.5.3.1.

old_theme_html: For semantics, see section 3.1.5.3.11.

expandWebPartPages: For semantics, see section 3.1.5.3.11.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

This method MUST return a multipart/mixed Multipurpose Internet Mail Extension (MIME) document (as specified in [RFC1341]) and the responses MUST be in URL Mode rather than HTML Mode.

Each part of the response MUST be one of the following three types: UrlArgs, DocInfo, or DocData. These types are defined and scoped to this section; they exist merely to provide convenient names for the concepts.

The response MUST begin with a UrlArgs part. After the first part, each part determines the type of the next part as illustrated in the following figure.

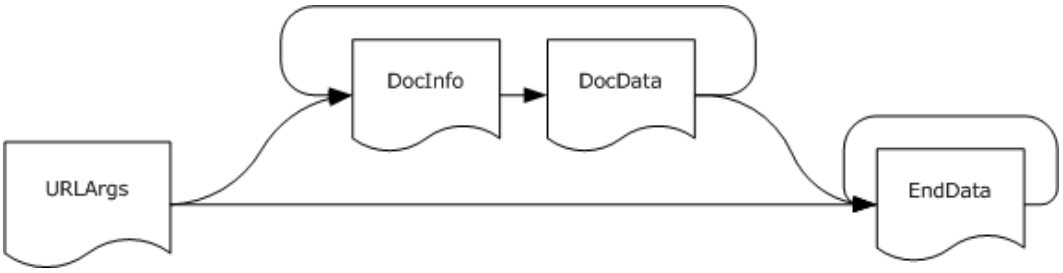

Figure 2: Type encoding sequence for POST

UrlArgs: The UrlArgs part of the response MUST have a Content-Type of application/x-www-form-urlencoded and MUST have a **METHOD-KEY-VALUE** (section 2.2.2.2.6) whose **REQUEST-NAME-STRING** (section 2.2.2.2.4) is "get documents" followed by a RET-NAME/RET-VAL pair whose **RPC-KEY-STRING** (section 2.2.2.2.5) is "current_time" and whose RPCVALUE is a **TIME** (section 2.2.2.1.6). The following is an example:

```
"method" VALSEP "get documents" ":" PROTOCOL-VERSION-STRING
ARGSEP "current_time" VALSEP TIME
```

The **UrlArgs** part MUST either be followed by a **DocInfo** part or the response MUST end.

DocInfo: The **DocInfo** part MUST be application/x-www-form-urlencoded and be a **DOCINFO** (section 2.2.2.2.12). Each **DocInfo** part MUST be followed by a **DocData** part.

DocData: The **DocData** part MUST have a Content-Type of application/octet-stream. It MUST contain the stream of the document that corresponds to the previous **DocInfo** part. Each **DocData** part MUST be followed by a **DocInfo** part or the response MUST terminate.

The mf-file-status metadata MUST be added to the **METADICT** (section 2.2.2.2.11) returned in the response. If it is nonzero, the remainder of the response SHOULD be discarded by the client. This is not considered metadata about the file, but rather metadata about the transport, which the client SHOULD examine and SHOULD remove before passing on the **METADICT** to higher layers.

3.1.5.3.13 get manifest

Used by the client to obtain a **manifest**, an XML document containing a copy of all or some of the information that specifies the **site** other than the contents of the documents on the site. A manifest contains information such as the names, locations, and metadata for the service, **folders**, and documents; the web structure; and the **list** schemas and data. It MAY also recursively include **subsites**.<31>

If a manifest is created with every option set, then taken in combination with a copy of the documents obtained using the get documents method, this forms a backup of the site.

A manifest created with only some options set can also be used by the client to create a template for new sites that use the same structure and settings. Detailed information within the manifest below the level of sites, documents, and folders is opaque to the client.

Parameters

options: A **STRING** (section 2.2.2.1.5) that contains the options that specify what to include in the manifest to be returned, as a comma-separated list of values. A manifest returned by the server MAY contain all or only certain types of information about the site, which the client specifies by using the *options* parameter.

The following option values are allowed.

Value	Meaning
everything	A shortcut for turning on every option. This requests a complete backup of the site in the manifest.
subwebs	Include subsites in the manifest.
structure	Include the web navigation structure in the manifest.
files	Include the names and metadata of the documents and folders in the manifest.
file_history	This option MUST be ignored by the server.
userlists	Include Lists and View schemas in the manifest.
list_data	Include the data associated with the included lists.
nontemplatizable_data	Include data that is not required to create a Site template from the included lists.
globallists	Include common userinfo with subscriptions and discussions data.
subscriptions	Include subscriptions information in the manifest.
discussions	Include web discussions information in the manifest.
userinfo	Include userinfo in the manifest, along with subscriptions or discussions.
webparts	Include the **Web Parts** information in the manifest.
security	This option MUST be ignored by the server.

Certain options are allowed only in combination with other options. The globallists and userinfo options MUST be included to use the subscriptions and discussions options. The userlists option MUST be included to use the list_data and nontemplatizable_data options.

Entry Point

FPAdminScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

subwebs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) that contains a list of the subsites in the current site. If the *options* parameter in the request included "subwebs", either explicitly or as part of the everything option, and the site has subsites, the server MUST include this list in the return value. Otherwise the server MUST exclude this from the return value.

document: The manifest XML file returned as the document stream that is sent after the </HTML> tag at the end of the standard response. The server MUST generate and return a manifest document for the site with the specified options to the client.

3.1.5.3.14 get theme

Used by the client to request the files for the named theme from the server. The server MUST return all the files for the theme as a multipart/mixed MIME **document**.<32>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

theme_name: A **STRING** (section 2.2.2.1.5) that contains the name of the theme to be returned. The client MUST send the name of the theme and the server MUST return the files associated with that theme name in the response.

Entry Point

FPAuthorScriptUrl

Return Values

This request MUST return a multipart/mixed MIME document (as specified in [RFC1341]) as described for the return value of the **get documents** (section 3.1.5.3.12) method, which contains all the theme files for the named theme in the response. This method is effectively a shortcut for the **get documents** method, with the *url_list* argument values constructed from the service-relative URLs of the files contained within the **folder** with theme named in the shared themes folder. The files MAY be in any order within the document.

3.1.5.3.15 get web struct

This method is used to get the internal web structure and **ELEMENT-IDs** (section 2.2.2.2.28) for the documents that make up that web structure. The web structure is conceptually a tree of nodes for each linked document, with links from each node to its parent and child nodes. The top-level or global pages displayed by the client in a navigation view are considered the child nodes of a virtual root node.<33>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

eidHead: The **ELEMENT-ID** of the root node that the server is required to use when retrieving the web structure. The **STRUCTURE-ELEMENTS** (section 2.2.2.2.32) are retrieved starting with this node and descending through the hierarchy.

This parameter MAY have a special value of 0 or 1, for sending the following predefined settings.

Value	Meaning
0	Start at the virtual root. If the *levels* parameter is set to get all levels, this retrieves the entire web structure.
1	Start with the first temporary element I.

includeHead: A **BOOLEAN** (section 2.2.2.1.3) value that indicates whether to include the node identified by the *eidHead* parameter in the results.

Value	Meaning
TRUE	The server MUST return the node specified by the *eidHead* parameter, along with the nodes specified by the *levels* parameter.
FALSE	The server MUST return only the nodes specified by the *levels* parameter.

levels: An **INT** (section 2.2.2.1.2) that identifies the number of levels to retrieve from the navigation structure store for the **site**. A value of -1 means that the server MUST retrieve all levels below the specified starting node. For a positive integer n, levels specifies that the server MUST retrieve n levels of child nodes. If the *includeHead* parameter is set to TRUE, 0 is legal and means that the server MUST retrieve just the specified node.

Entry Point

FPAuthorScriptUrl

Return Values

elements: A VECTOR-STRUCTURE-ELEMENT that contains a list of the **STRUCTURE-ELEMENT** data for each returned node in the web structure requested.

3.1.5.3.16 getDocsMetaInfo

The getDocsMetaInfo request is used by the client to retrieve metadata for the **documents** and **folders** in the current **site**. The **getDocsMetaInfo** request is similar in function to the **list documents** (section 3.1.5.3.20) request except that it only retrieves metadata for the documents or folders specified through the *url_list* parameter.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

listHiddenDocs: A **BOOLEAN** (section 2.2.2.1.3) value that specifies whether the client requests hidden documents in a site included in the documents to be listed. If TRUE, the server SHOULD list hidden documents; if FALSE, the server MAY leave hidden documents out of the list returned. The server SHOULD default this value to TRUE if it is not sent by the client.

listLinkInfo: For semantics, see *listLinkInfo* in section 3.1.5.3.1. The server MUST default this value to TRUE if it is not sent by the client.

url_list: A **VECTOR-STRING** (section 2.2.2.2.1) list of service-relative URLs for which the client requires information. If *url_list* is not sent or empty, it MUST be treated as though it is a request for the root of the site.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

document_list: A **DOCUMENT-LIST-RETURN-TYPE** (section 2.2.2.2.13) that specifies the name and metadata for the set of documents requested in the *url_list* parameter.

urldirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) that contains names and metadata for the folders and **subsites** in the current site requested in the *url_list* parameter. The server MUST return urldirs if url_List contains at least a folder.

failedUrls: A **VECTOR-URL-STRING** (section 2.2.2.2.3) list of service-relative URLs specifying which of the entries in the *url_list* parameter of the request failed to have the corresponding **METADICT** (section 2.2.2.2.11) applied.

3.1.5.3.17 html-table add row

Used by the client to add a named task-list filerow to a specified task-list file. Clients MAY use this method to update the task-list files.<34>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

url: A **URL-STRING** (section 2.2.2.2.3) with the service-relative URL of the task-list file that contains the Task Status table to be modified.

newRow: A **VECTOR-STRING** (section 2.2.2.2.1) that contains the encoded <TR> data for the new row of the Tasks table that is being changed on the server, including the user who made the status change, the file that has a changed status, and the new status.

Entry Point

FPAuthorScriptUrl

Return Values

newRowId: This **INT** (section 2.2.2.1.2) contains the **newRowId** for the newly added row.

3.1.5.3.18 html-table change row

Used by the client to change a named row in a specified task-list file that contains a Task Status table. Clients MAY use this to update the task-list files.<35>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

url: For semantics, see section 3.1.5.3.17.

newRow: For semantics, see section 3.1.5.3.13. The ordinal position of the row to be changed is found in the first <TD> element of the encoded <TR> passed as the argument for *newRow*.

Entry Point

FPAuthorScriptUrl

Return Values

newRowId: This **INT** (section 2.2.2.1.2) contains the row identifier for the newly changed row.

3.1.5.3.19 html-table remove row

Used by the client to remove a particular row in a specified task-list file. Clients MAY use this to update the task-list files.<36>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

url: For semantics, see section 3.1.5.3.17.

rowId: This parameter is the ordinal position of the row in the Task Status table that the server MUST remove.

Entry Point

FPAuthorScriptUrl

Return Values

rowId: This **INT** (section 2.2.2.1.2) contains the row identifier for the deleted row.

3.1.5.3.20 list documents

The **list documents** request is used by the client to request a list of files, **folders**, and **sites** that is contained in a given folder.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

folderList: The *folderList* parameter is a **METADICT** (section 2.2.2.2.11) containing the names of folders as the keys, and the corresponding time stamps of type **TIME** (section 2.2.2.1.6) for the client's cached metadata for those folders as the values. The client SHOULD send the time stamps for folders' metadata if it has cached this information. If a folder's name and time stamp are missing from the *folderList* parameter, the server MUST return that folder's contents with all available metadata. Otherwise, the server MUST return all available metadata for only those files, folders, and subsites which have changed since the time stamp recorded in the *folderList*. For files, folders, and subsites which have not changed, the server SHOULD return an empty **METADICT** to indicate that the client's cache is still valid. The client MUST interpret an empty **METADICT** as an indication of validity.

The following example of a value for the *folderList* parameter requests metadata for all files in the root folders of the site and in the images folders that have a time stamp later than the specified date and time. For other files in these folders, only the file name and an empty **METADICT** are included in the response. For all other folders within the *initialUrl* parameter, all available metadata MUST be returned.

```
[;TX|06 Apr 2006 20:03:02 -0000;images;TX|05 Apr 2006 20:03:02 -0000]
```

initialUrl: A **URL-STRING** (section 2.2.2.2.3) that contains the URL of the folder from which **documents** are listed. This MUST be a service-relative URL.

listBorders: A **BOOLEAN** (section 2.2.2.1.3) value that specifies whether the contents of the shared borders folders that contains shared border pages SHOULD be listed. If TRUE, the contents SHOULD be listed; if FALSE, they SHOULD NOT be listed.

listChildWebs: A **BOOLEAN** value that specifies whether the server response SHOULD include the names of the child site folders in the **urldirs** return value. If TRUE, and the *listFolders* parameter is also sent as TRUE, the names SHOULD be included; if FALSE, they SHOULD NOT be included. If the client sends this parameter as TRUE, the client SHOULD also send the *listFolders* parameter as TRUE. The server SHOULD ignore a TRUE value for this parameter if the *listFolders* parameter is not also TRUE.

listDerived: A **BOOLEAN** value that specifies whether the list of files in the derived documents folder SHOULD be included in the response. If TRUE, the list of files SHOULD be included as part of the response; if FALSE, they SHOULD NOT be included.<37>

listExplorerDocs: A **BOOLEAN** value that specifies whether task-list files (_vti_pvt/_x_todo.htm and _vti_pvt/_x_todoh.htm) are listed. If TRUE, the files MAY be listed. If FALSE, they MUST NOT be listed. Servers SHOULD ignore this value if received.

listFiles: A **BOOLEAN** value that specifies whether the client requests information about the files in each folder that appears in the response. If TRUE, the server SHOULD include the **document_list** return value; if FALSE, the server SHOULD exclude the **document_list** return value. By default, the server SHOULD set this value to TRUE.

listFolders: A **BOOLEAN** value that specifies whether the server response SHOULD include the names and metadata of folders under the URL specified by the *initialUrl* parameter. If TRUE, the names and metadata of the folders SHOULD be included; if FALSE, they SHOULD NOT be included. By default, the server SHOULD set this value to TRUE.

listHiddenDocs: A **BOOLEAN** value that specifies whether hidden documents in a site SHOULD be listed. If TRUE, the documents SHOULD be listed; if FALSE, they SHOULD NOT be listed.

listIncludeParent: A **BOOLEAN** value that specifies whether an entry for the *initialUrl* field SHOULD be included in the server response. If TRUE, the entry SHOULD be included; if FALSE, it SHOULD NOT be included.

listLinkInfo: For semantics, see section 3.1.5.3.1. By default, the server SHOULD set this value to TRUE.

listRecurse: A **BOOLEAN** value that specifies whether the server response SHOULD recursively list the subfolders of folders under the URL specified by the *initialUrl* parameter. If TRUE, the subfolders SHOULD be listed; if FALSE, they SHOULD NOT be listed. The default value is TRUE unless the client specifies otherwise. If *listRecurse* is set, all child folders of that folder SHOULD be taken into account; otherwise, only the immediate child folders SHOULD be considered. By default, the server SHOULD set this value to TRUE.

listThickets: A **BOOLEAN** value that specifies whether **thicket** supporting files and folders SHOULD be included in the server response. If TRUE, the supporting files and folders SHOULD be included; if FALSE, they SHOULD NOT be included. By default, the server SHOULD set this value to TRUE.

platform: A **STRING** (section 2.2.2.1.5) that specifies the operating system of the client that controls the listing of any open **web bot** (custom component). If the field is missing or empty, the server MUST NOT include the **bot_list** return value in the response. If the field contains data, information about any open web bot on the server MAY be included. This parameter MUST NOT be passed by clients conforming to the FrontPage Server Extensions: Website Management Protocol.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

document_list: For semantics, see **document_list** in section 3.1.5.3.1. The server MUST omit this return value if the *listFiles* parameter was sent as FALSE. If this value is returned, the server MUST list the names and metadata for the requested set of documents specified by the parameters of the request.

bot_list: A **DOCUMENT-LIST-RETURN-TYPE** (section 2.2.2.2.13) that MUST NOT be included when the *platform* parameter is empty (including when the parameter is not sent). Because clients conforming to the FrontPage Server Extensions: Website Management Protocol MUST NOT send this parameter, the server need not support this return value. The server MAY<38> return an empty **bot_list** if it wants to partially support this value.

urldirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) that contains the names and metadata for the specified folders and root directories of subsites. The server MUST omit this return value if the *listFolders* parameter was sent as FALSE. If this value is returned, the server MUST enumerate the folders and sites specified by the parameters of the request.

3.1.5.3.21 list themes

The client uses the **list themes** method to obtain a list of names of themes available on the **site** and metadata about each of the available themes.<39>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

lcid: An **INT** (section 2.2.2.1.2) that contains the **LCID** that the server SHOULD use to generate the text of the Theme titles in the returned **document_list**. By default, the server MUST return the default Theme title in the returned **document_list**.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. The server MUST return a **list** of the theme names and the available metadata about each named theme found in the shared themes folder of the site.

3.1.5.3.22 list versions

Used by the client to obtain a list of the versions of a particular **document** on the server in its source control system.<40>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

document_name: For details, see *document_name* in section 3.1.5.3.1. The client MUST send this parameter as the service-relative URL of the document for which the server MUST return a list of versions.

Entry Point

FPAuthorScriptUrl

Return Values

version_list: A **VECTOR-DOCINFO** (section 2.2.2.2.12) with metadata information for each version of the document stored on the server under source control.

3.1.5.3.23 move document

The **move document** request is used by the client to change the URL of a selected **document** or **folder** in the **site**.

Note Moving a document will change its URL.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

docopy: A **BOOLEAN** (section 2.2.2.1.3) value that specifies if the move document request SHOULD copy or move a file to the destination. If TRUE, the file SHOULD be copied; if FALSE, the file SHOULD be moved. The default value is FALSE.

newUrl: A **URL-STRING** (section 2.2.2.2.3) that specifies the new service-relative URL for the document or folder whose URL is to be changed. The client MUST send this parameter.

oldUrl: A **URL-STRING** that specifies the original service-relative URL for the document or folder whose URL is to be changed. The client MUST send this parameter.

put_option: A set of **PUT-OPTION** flags that describe how the operation SHOULD behave as specified in section 2.2.2.2.18. In particular, the server MUST overwrite an existing file or folder if, and only if, the "overwrite" flag is added.

rename_option: A **RENAME-OPTION** value that specifies server behaviors during the rename or copy operation, as specified in section 2.2.2.2.19.

url_list: A **VECTOR-URL-STRING** (section 2.2.2.2.3) list of service-relative URLs of documents whose links SHOULD be considered for **link fixup** purposes. This is a hint passed from the client to the server. Servers that implement link fixup SHOULD NOT rely on the client sending the correct list. Clients that conform to the FrontPage Server Extensions: Website Management Protocol MUST send an empty list (either explicitly or by omitting the parameter and taking the default). Servers SHOULD ignore this parameter.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

oldUrl: A **URL-STRING** that specifies the original service-relative URL for the document whose name or folder has changed.

newUrl: A **URL-STRING** that specifies the new service-relative URL for the document whose name or folder has changed.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. The server MUST return the set of documents whose metadata has changed because of link fixup as a result of the move.

moved_docs: A **DOCUMENT-LIST-RETURN-TYPE** (section 2.2.2.2.13) that contains the names and metadata for all the documents which have been copied or moved. If no documents have been copied or moved, the server MUST return an empty value for this result.

moved_dirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) of the service-relative URLs and metadata for all folders that have moved as a result of this method. If no folders have moved, the server MUST return an empty value for this result.

3.1.5.3.24 open service

The **open service** request is used to provide **site** metadata to the client.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

effective_protocol_version: This parameter is deprecated. For semantics, see *effective_protocol_version* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

service: A **SERVICE-RETURN-TYPE** (section 2.2.2.2.14) that specifies the service name and site metadata.

3.1.5.3.25 put document

The **put document** request is used by the client to write a single **document** to a specified service-relative URL in an existing **site**.

Parameters

service_name: This parameter is deprecated. For semantics, see *service_name* in section 3.1.5.3.1.

document: A **DOCINFO** (section 2.2.2.2.12) that specifies the service-relative URL and metadata of the document to write to the site. The client MUST send this parameter.

put_option: A set of **PUT-OPTION-VAL** flags that describe how the operation behaves; see **Put-Option** (section 2.2.2.2.18) for specific semantics for the options.

comment: A **STRING** (section 2.2.2.1.5) that provides a checkin comment for the file being uploaded.

keep_checked_out: A **BOOLEAN** (section 2.2.2.1.3) value that SHOULD determine a specified document's behavior in source control. If TRUE, the document SHOULD be checked in to source control and immediately checked back out; if FALSE, the document SHOULD be checked in. The server MUST treat this as equivalent to the "checkout" **PUT-OPTION-VAL**, as specified in section 2.2.2.2.18.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

document: A **DOCINFO** that contains the name and metadata of the document as it was saved. Although this return value is called "document", it does not contain the document stream. The server updates the metadata when the document is saved. The server MUST return the updated metadata.

3.1.5.3.26 put documents

The put documents request is used by the client to write multiple files to a **site**. This request SHOULD be used when a higher level wants to save a **document** that contains other files (such as graphics) from an application directly to a site **folder**.

This request is different from other FrontPage Server Extensions: Website Management Protocol requests because it sends multiple streams. The following details show how the server MUST parse the method.

The HTTP POST MUST be a multipart/mixed MIME document. Each part of the POST MUST be one of the following four types: **UrlArgs**, **DocInfo**, **DocData**, or **End**. These types are defined and scoped to this section; they exist merely to provide convenient names for the concepts.

The request MUST begin with a **UrlArgs** part. After the first part, each part determines the type of the next part, as illustrated in the following figure.

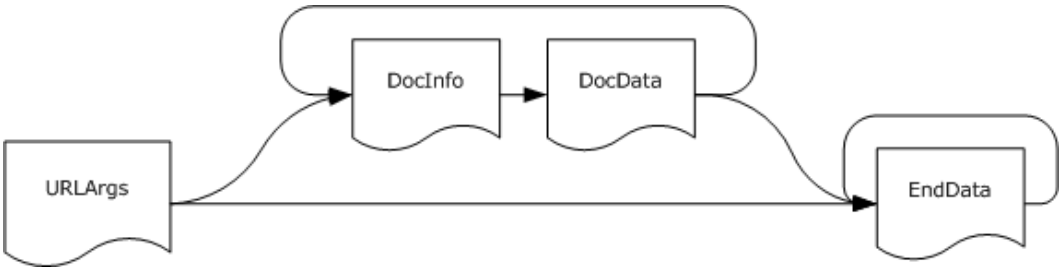

Figure 3: Type encoding sequence for POST

UrlArgs: The **UrlArgs** part MUST have a Content-Type of application/x-www-form-urlencoded (as specified in [RFC2616] section 14.17) and MUST be parsed like a normal method request. The **REQUEST-NAME-STRING** (as specified in [RFC4234] and section 2.2.2.2.4) MUST be "put documents" and it accepts the list of parameters given in the following Parameters section. The server SHOULD fail with a "client-too-old" error if the client is an earlier version than the server supports, just like any other method.

If the next part exists, it MUST be **DocInfo** with a Content-Type of application/x-www-form-urlencoded (as specified in [RFC2616] section 14.17) or it MUST be End with a Content-Type of text/html.

DocInfo: The server SHOULD parse the **DocInfo** part as a **DOCINFO** (for details, see section 2.2.2.2.15). The next MIME part MUST be **DocData**.

DocData: A **DocData** part MAY have any Content-Type and MUST be the stream corresponding to the **DOCINFO** sent in the prior **DocInfo** MIME part. If the next part exists, it SHOULD be **DocInfo** if its Content-Type is application/x-www-form-urlencoded (as specified in [RFC2616] section 14.17) or SHOULD be End if its Content-Type is text/html, also specified in [RFC2616].

End: An **End** part MUST be ignored by the server. Any subsequent part SHOULD also be considered to be an **End** part.

The server SHOULD accept and ignore a **DocInfo/DocData** pair if the URL in the **DOCINFO** is empty. Clients SHOULD avoid doing this because it wastes bandwidth.

If the "atomic" **PUT-OPTION-VAL** is specified as defined in section 2.2.2.2.18, the server SHOULD either succeed in storing all the documents or not store any of them. If the client does not request the "atomic" option, or if the server does not honor the option, and if the server is unable to store a document, documents in the request before the one that failed MUST be stored, and documents after the one that failed MUST NOT be stored.

Parameters

service_name: This parameter is deprecated. For semantics, see *service_name* in section 3.1.5.3.1.

listFiles: A **BOOLEAN** (section 2.2.2.1.3) value that specifies whether the docs return value will be included in the response. The server MUST include the docs return value if this parameter is TRUE.

listLinkInfo: For semantics, see *listLinkInfo* in section 3.1.5.3.1.

put_option: A set of **PUT-OPTION-VAL** flags that describe how the operation behaves; see section 2.2.2.2.18 for specific semantics for the options.

time_tokens: A VECTOR-TIME (section 2.2.2.2.1). If this argument is sent, it MUST either be empty (which is equivalent to not sending it) or it MUST contain a **TIME** (section 2.2.2.1.6) entry for each document that will be sent in the remainder of the request. If this argument contains **TIME** values, and the put_option argument includes "edit", the server SHOULD check this value when performing the check for modifications, in addition to checking the **vti_timelastmodified** (section 2.2.2.3.82)

metakey sent with the metadata for each document, as described in **Put-Options** for "edit" in section 2.2.2.2.18.<41>

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

docs: A **VECTOR-DOCINFO** (section 2.2.2.2.12) that contains information about the saved documents. This value MUST NOT be included in the response if *listFiles* is passed as FALSE.

error-index: This **INT** (section 2.2.2.1.2) is returned only in error conditions for a single document rather than the transfer in general (for example, because of a time stamp mismatch, a document exists, or a document is checked out). If present, it MUST be the zero-based index of the document that the server was unable to store.

document: This **DOCINFO** MUST be returned if, and only if, error-index is also returned. This SHOULD indicate the document URL and metadata for the document that could not be uploaded.

3.1.5.3.27 put manifest

Used by the client to restore a **manifest**, as the XML document returned by the **get manifest** (section 3.1.5.3.13) method. The manifest contains a copy of all or some of the information that specifies a **site** other than the contents of the documents on the site.<42>

A complete manifest, along with a copy of the documents within the site, constitutes a backup of the site. A client can restore such a backup using this method. A client can also use the manifest as a template for a creating new site's settings and structure using this method. Detailed information within the manifest below the level of sites, documents, and **folders** is opaque to the client.

Restoring a site backup requires multiple method calls. Server state MUST be restored, along with the document content, and because the documents are restored with a separate call to the **put documents** (section 3.1.5.3.26) method, this cannot be completed with a single call to put manifest. Instead, restoration occurs in two passes for each site in the manifest.

First Pass, Phase One

The client MUST call put manifest with the "first pass" option. The prefix option MAY be used to specify that the server perform the **put manifest** operation on the named **subsites** within the manifest. The *url_renames* parameter and the *guidmap* parameter MUST be empty. The server performs the following operations:

1. Applies the site template to the site.
2. Restores general site settings.
3. Restores **Lists** and List data.
4. Creates the folder structure and 0-byte temporary files for **document library** files.

When the server has completed these operations, the server reports the results to the client, along with information the client MUST use to complete the restoration.

First Pass, Phase Two

The client then uses the skip urls, renamed urls, and moved urls results returned by the server to modify the list of folders and files it plans to upload to the server. The client can then use any combination of the **put document** (section 3.1.5.3.25), **put documents** (section 3.1.5.3.26), **create url-directory** (section 3.1.5.3.10), and **create url-directories** (section 3.1.5.3.9) methods to upload the files and folders.

The files listed in the **skip urls**, **renamed urls**, and **moved urls** results returned by the server SHOULD be skipped, renamed, or moved accordingly by the client during the upload phase. The server creates most folders during the first pass, but does not create folders within Document Libraries. The client MUST first create folders within document library lists during this phase, and then upload the files.

Second Pass

After completing the uploads, the client MUST call **put manifest** with the "second pass" option, and with the parameters set using the results from the first pass. The server performs the following operations during the second pass:

1. Performs link repair in documents.
2. Finalizes navigation structure.
3. Restores **Web Parts**.
4. Restores discussions.
5. Restores subscriptions.
6. Updates item IDs.

When the manifest has been applied to the site, it MAY be applied recursively to each subsite within that site, if any. This process follows the same outline, using the *prefix* parameter to specify which subsite the **put manifest** method is applied to.

Parameters

options: A **STRING** (section 2.2.2.1.5) that contains options describing which actions to perform during the **put manifest** method. This MAY be one of two mutually exclusive options, indicated by the presence of one of the following strings.

Value	Meaning
"first pass"	Perform filename analysis and preparation for the put manifest action. If this option is set, the **url_renames** and **guidmap** options are ignored by the server.
"second pass"	Perform the final pass of the **put manifest** action.

prefix: A **URL-STRING** (section 2.2.2.2.3) that contains the service-relative URL of a subsite to apply the manifest to. This value MUST be empty to restore the root site in the manifest.

url_renames: A **DICT** (section 2.2.2.2.10) of URL-to-URL mappings for renaming files and folders. The client MUST NOT set this parameter if the first pass option is set. The values to use with this parameter are passed to the client in the renamed urls return value of the put manifest response returned at the end of the first pass.

The client MUST send that data back to the server in this parameter for the second pass. This option MUST be ignored by the server if the "first pass" option is set. The server MUST use this mapping to rename URLs in the site if the "second pass" option is set.

guidmap: A **VECTOR-STRING** (section 2.2.2.2.1) containing **GUID** strings which, taken as key-value pairs, represent an update mapping for GUIDs used as IDs for items in the manifest showing mapping onto their new IDs in the resulting site.

The client MUST NOT send this parameter if the "first pass" option is set. If the "second pass" option is set, the client MUST send as the content of this parameter the result of the **guidmap** return value returned by the "first pass" call to put manifest. This option MUST be ignored by the server if the "first pass" option is set. The server MUST use this mapping to update the IDs of documents and folders in the site if the "second pass" option is set.

filelist: A **VECTOR-STRING** that specifies a list of service-relative names of the folders and files within the manifest. When the *options* parameter is set to "first pass", this MUST be a list of all the files and folders contained within the manifest. When the *options* parameter is set to "second pass" the *filelist* parameter is used to specify all of the files within the manifest that contain the **vti_linkbars** (section 2.2.2.3.46) metakey.

If a file contains the **vti_linkbars** metakey, and it is listed in the **renamed urls** section at the end of the first pass of the **put manifest** response, that file MUST be handled differently from other files. Instead of passing the original file and folder name to this parameter in the second pass, the client SHOULD pass the new file and folder name recommended by the server.

Files listed in the **skip urls** section at the end of the first pass MUST NOT be listed in the *filelist* parameter for the second pass.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

skip urls: A **VECTOR-STRING** that contains a list of the URLs of the documents and folders which the server generates automatically. This list specifies the files the client MUST NOT upload to the server between the first pass and second pass calls to put manifest when restoring a site.

The client MUST NOT include these URLs in the *filelist* parameter during the second pass of the **put manifest** operation. The server MUST specify this result if the *options* parameter is "first pass". The server MAY include this result if the *options* parameter is "second pass".

renamed urls: A **DICT** containing a list of key-value pairs consisting of the item's original URL and a new URL for that item. The server recommends that these files and folders be renamed.

These files and folders SHOULD be uploaded from the client using the new recommended names. If the same file name is listed in the **skip urls** and **renamed urls** sections, the client SHOULD NOT upload the file.

This return value section is used by the *url_renames* parameter during the second pass. The server MAY create this list for files and folders which the server determines need to be renamed from the names originally specified in the manifest. The server MUST specify this result if the *options* parameter is "first pass". The server MAY include this result if the *options* parameter is "second pass".

moved urls: A **DICT** that contains a list of key-value pairs consisting of the original file URL and the URL the server requests the client move the file to. The client SHOULD upload the files in this list to the new locations specified. The server MAY create this list for files which the server determines need to be moved from the location originally specified in the manifest.

The server MUST specify this result if the *options* parameter is "first pass". The server MAY include this result if the *options* parameter is "second pass".

guidmap: A VECTOR-STRING containing GUID strings which, taken as key-value pairs, represent an update mapping for GUIDs used as IDs for items in the manifest mapping onto their new IDs in the resulting site.

The client MUST include this result in the *guidmap* parameter to the second pass call to **put manifest**. The server MAY create this list for items which the server determines need to have their IDs updated from the IDs originally specified in the manifest.

The server MUST specify this result if the *options* parameter is "first pass". The server MAY include this result if the *options* parameter is "second pass".

successes: A **VECTOR-STRING** that contains success messages which occur during the **put manifest** method processing by the server. The server MAY record such messages and report them here. The client MAY log these messages or display them to the user.

failures: A **VECTOR-STRING** that contains error messages for each of the failures that occur during the **put manifest** method processing by the server. The server MUST record such messages and report them here. The client MAY log these messages or display them to the user.

urldir: A **URL-DIRECTORY** (section 2.2.2.2.16) that contains the service-relative name and metadata for the subsites. This return value MUST be set by the server only if the *options* parameter was "first pass" and the *prefix* parameter was set in the request.

3.1.5.3.28 put theme

The client uses the **put theme** method<43> to upload a set of theme files and associated metadata to the server. The theme MUST be encoded as a multipart/mixed MIME **document** in the same format the server returns for the **get theme** (section 3.1.5.3.14) method.

Parameters

This method requires no parameters. Instead, the theme is uploaded as a multipart/mixed MIME document as in the **put documents** (section 3.1.5.3.26) method, but with a nonstandard MIME type of multipart/mixed/theme.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

3.1.5.3.29 put web struct

This method is used to update the web structure and element identification for the **documents** that make up that structure on the server.<44>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

changes: The changes parameter is a **VECTOR-STRUCTURE-MODIFICATION** (section 2.2.2.2.33) that contains a list of the web structure elements to be changed in the **site**. The server MUST apply the changes to its copy of the web navigation structure to the nodes with matching **ELEMENT-IDs** (section 2.2.2.2.28) and perform fixup on the web navigation links in all documents that are affected by the change.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

failedMod: A **STRUCTURE-MODIFICATION** that contains the first **VECTOR-STRUCTURE-MODIFICATION** element in the *changes* parameter that failed to update the navigation structure. This value MUST only be returned if a web structure modification failed. If this value is returned, the server MUST also return a STATUS (section 2.2.2.2.17) with the error caused by the failure.

elements: A **VECTOR-STRUCTURE-ELEMENT** (section 2.2.2.2.32) that contains a list of the **STRUCTURE-ELEMENT** data for each successfully changed node in the web structure. This value MUST only be returned if no web structure modification failed and there are changed nodes in the web structure.

docs: A **VECTOR-DOCINFO** (section 2.2.2.2.12) that specifies the service-relative URLs and metadata of the documents that were updated by the changes to the web structure. The server MUST return this value with a (potentially empty) list of documents that were updated.

3.1.5.3.30 recalc control

This method is sent by a FrontPage client at the end of a publish operation. The server MAY use this method request as a signal to perform updates.<45>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

recalc_op: This **STRING** (section 2.2.2.1.5) parameter has the following options:

- suspend
- resume

These flags are no longer used with their original meanings in the FrontPage Server Extensions: Website Management Protocol. Clients MAY send any of these values as the *recalc_op* argument. If an argument is present, the server MUST validate the argument and return an error if it contains something other than one of the listed options. The server MUST ignore an argument value of "suspend". The server MAY use an argument value of "resume" to perform housekeeping operations such as deleting unused themes from the **site**.

Entry Point

FPAuthorScriptUrl

Return Values

None.

3.1.5.3.31 remove documents

The **remove documents** request is used by the client to delete specific **documents** or **folders** from the **site**.

Parameters

service_name: This parameter is deprecated. For semantics, see *service_name* in section 3.1.5.3.1.

url_list: A **VECTOR-URL-STRING** (section 2.2.2.2.3) list of service-relative URLs that the client wants to be deleted. The server SHOULD delete the URLs listed here, subject to authorization checks.

time_tokens: If present and nonempty, this VECTOR-TIME (section 2.2.2.2.1) argument lists the **vti_timelastmodified** (section 2.2.2.3.82) metakey values as known on the client for the corresponding documents in the *url_list* argument. If the **VECTOR-TIME** is empty or the parameter is not present, the server MUST ignore this parameter; otherwise, it MAY refuse to delete documents in which the time stamp does not match the actual **vti_timelastmodified** value as known on the server <46>.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

failed_dirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) that specifies the name and metadata for the folders that failed to be removed. The server MUST respond with a (potentially empty) list of folders that could not be removed.

failed_docs: A **VECTOR-DOCINFO** (section 2.2.2.2.12) that specifies the name and metadata for the documents that failed to be removed. The server MUST respond with a (potentially empty) list of documents that could not be removed. The server SHOULD send empty **METADICTs** (section 2.2.2.2.11) in this return value.

removed_dirs: A **VECTOR-URL-DIRECTORY** that contains the name and metadata for the folders that were removed. The server SHOULD send empty **METADICTs** in this return value.<47>

removed_docs: A **VECTOR-DOCINFO** that contains the name and metadata for the documents that were removed. The server SHOULD send empty **METADICTs** in this value <48>.

3.1.5.3.32 remove service

Used by the client to remove the specified **site**. The server SHOULD NOT delete the site if there are any **subsites** present in the site. To delete a site with subsites, the subsites SHOULD first be deleted. <49>The server SHOULD NOT delete the root site of the **site collection**.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1. The client MUST send this server-relative **URL-STRING** (section 2.2.2.2.3) parameter to specify the site to remove.

flags: A **STRING** (section 2.2.2.1.5) that MUST be "default".

Entry Point

FPAdminScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

3.1.5.3.33 rename service

Changes the root URL of a **site**.

Parameters

service_name: See *service_name* in section 3.1.5.3.1. The client MUST send this server-relative **URL-STRING** (section 2.2.2.2.3) parameter to specify the site to rename.

newName: A **STRING** (section 2.2.2.1.5) with the new root URL for the site. The server MUST attempt to change the root URL of the site to the value of this argument.

flags: Contains an **INT** (section 2.2.2.1.2) that MUST be ignored by the server but MAY be sent by the client and SHOULD equal 0.

Entry Point

FPAdminScriptUrl

Return Values

service: A **SERVICE-RETURN-TYPE** (section 2.2.2.2.14) that contains the new **server-relative URL** and all the metadata available to the client for the site after application of the rename service method.

3.1.5.3.34 rename url

Modifies the specified **documents** or all documents on a **site** to have existing links to a specified original URL updated to point to a new URL.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

oldUrl: A **URL-STRING** (section 2.2.2.2.3) that specifies the original URL to be changed in the documents specified by the *url_list* argument. The client MUST send this parameter, which MAY be an absolute URL, a **server-relative URL**, or a service-relative URL.

newUrl: A **URL-STRING** that specifies the new URL that all references to the *oldUrl* argument are to be fixed up to, in the documents specified by the *url_list* argument. The client MUST send this parameter, which MAY be an absolute URL, a server-relative URL, or a service-relative URL.

url_list: For semantics, see section 3.1.5.3.1. This is a list of the documents which the client indicates the server MUST update by changing link references to the URL specified in the *oldUrl* argument into link references to the URL specified in the *newUrl* argument. If this parameter is omitted or empty, the server MUST NOT make changes to any documents.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

message: For semantics, see **message** in section 3.1.5.3.1.

oldUrl: A **URL-STRING** that contains the original URL changed in link references within the specified documents.

newUrl: A **URL-STRING** that contains the new URL that link references within the specified documents are changed to.

document_list: For semantics, see **document_list** in section 3.1.5.3.1. This lists the documents that have been changed as a result of the **link fixup** process.

3.1.5.3.35 replace web struct

This method is used to replace the web structure and element identification for the **documents** that make up that structure. The server MUST clear the existing web structure and replace it with the structure elements passed in by the client in the *elements* parameter.<50>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

elements: The *elements* argument is a **VECTOR-STRUCTURE-ELEMENT** (section 2.2.2.2.32) which contains a list of the web structure elements to be set in the **site**. The server MUST clear its existing web structure and replace it with this argument. The server MUST perform **link fixup** in the web navigation links in all documents on the site affected by the change to enable the new web structure.

Entry Point

FPAuthorScriptUrl

Return Values

None.

3.1.5.3.36 server version

The **server version** request is to be used by the client to request the version of the server extensions in use on the server.

Parameters

The argument list for this request SHOULD be empty. The server MUST ignore any parameters sent.

Entry Point

FPShtmlScriptUrl

Return Values

server_version: A **VERSION** (section 2.2.2.2.9) that specifies the current version of the server (not the effective protocol version). The server MUST respond with its actual version, which might be larger than the effective protocol version in the **PROTOCOL-VERSION-STRING** (section 2.2.2.2.2) built in to all the FrontPage Server Extensions: Website Management Protocol responses, as specified in section 2.2.2.2.7.

source_control: An **INT** (section 2.2.2.1.2) that indicates that the server supports the **checkout document** (section 3.1.5.3.7) and **uncheckout document** (section 3.1.5.3.40) requests. This value MUST equal 0 if the server does not support these requests; otherwise, this value MUST equal 1. Nonzero values other than 1 are reserved, but the client MUST interpret any nonzero value as if it were the value 1.

As with other methods, the effective protocol version negotiated by using the mechanism specified in section 3.1.3.2 SHOULD be returned in the **METHOD-KEY-VALUE** (section 2.2.2.2.6) element of the response.

3.1.5.3.37 set service meta-info

This method is used by the client to set metadata associated with the **site** on the server.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

meta_info: For semantics, see *meta_info* in section 3.1.5.3.1. This parameter is a **METADICT** (section 2.2.2.2.11) that contains all of the metadata to apply to the site. The client MAY update only some of the metadata in the site, leaving the remainder as is, by including in the **METADICT** only the metadata to be changed. The server MUST apply the metadata changes to the site that the client has read-write access to, and SHOULD ignore any metadata changes that the client does not have read-write access to.

Entry Point

FPAuthorScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** contains all the metadata available to the client for the site after application of the update.

3.1.5.3.38 set source control

This method is deprecated. If this method is called, and the *project* argument is sent with any value, it returns the metadata for the site.<51> <52>

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

project: The parameter MUST be a **STRING** (section 2.2.2.1.5) or a **METADICT** (section 2.2.2.2.11) that indicates the name of the source control project to use on the server. The client MUST send this parameter and SHOULD send the source control project name of the site, specified by its **vti_sourcecontrolproject** (section 2.2.2.3.74) metakey value. The server MAY ignore this parameter.

add_existing_pages: The parameter MUST be a **BOOLEAN** (section 2.2.2.1.3) that indicates whether to add existing pages into source control. The server MUST ignore this parameter.

Entry Point

FPAdminScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** contains all the metadata available to the client for the site.

3.1.5.3.39 setDocsMetaInfo

The **setDocsMetaInfo** request is used by the client to request that a list of **documents**, **folders**, and **subsites** have a corresponding list of **METADICTs** (section 2.2.2.2.11) applied to each entry. This is a way of combining several calls into a single call.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

listHiddenDocs: This argument MUST be ignored by the server.

listLinkInfo: For semantics, see *listLinkInfo* in section 3.1.5.3.1. The server MUST default this value to TRUE if it is not sent by the client.

url_list: For semantics, see *url_list* in section 3.1.5.3.1. This is the list of documents, folders, and subsites that the corresponding **METADICTs** in the *metaInfoList* argument MUST be applied to by the

server. If *url_list* is missing or empty, the server SHOULD default to applying the metaInfoList argument to the root of the **site**.

metaInfoList: A VECTOR-METADICT of **METADICTs** to associate with each of the entries in the *url_list* in order. This list MUST contain at least one entry, and exactly as many entries as the *url_list* if any are specified.

errorFlags: A **STRING** (section 2.2.2.1.5) that contains an ERROR-OPTION (section 2.2.2.2.20) value indicating the client preference for server behavior on errors. This parameter MAY be omitted, in which case the server MUST default to "stopOnFirst" behavior.

listFiles: A **BOOLEAN** (section 2.2.2.1.3) value that specifies whether the client requests metadata about all the files, directories, and site that appear in the response. If TRUE, the server MUST include the **document_list** return value; if FALSE, the server MUST exclude the **document_list** and **urldirs** return values.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

document_list: A **DOCUMENT-LIST-RETURN-TYPE** (section 2.2.2.2.13). The server MUST omit this return value if the *listFiles* parameter was sent as FALSE. If this value is returned, the server MUST list the names and updated metadata for the requested set of documents specified by the *url_list* parameter of the request.

failedUrls: A **VECTOR-URL-STRING** (section 2.2.2.2.3) list of service-relative URLs specifying which of the entries in the *url_list* parameter of the request failed to have the corresponding **metadict** applied.

urldirs: A **VECTOR-URL-DIRECTORY** (section 2.2.2.2.16) that contains the names and metadata for folders and root directories of subsites. The server MUST omit this return value if the *listFiles* parameter was sent as FALSE. If this value is returned, the server MUST enumerate the folders and subsites specified by the *url_list* parameter of the request.

3.1.5.3.40 uncheckout document

The uncheckout document request is used by the client to reverse a long-term checkout of a file from source control. If the file has changed since it was checked out, those changes are reverted. This request is also used to release a short-term checkout, in which case changes are not reverted.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

document_name: For semantics, see *document_name* in section 3.1.5.3.1. The client MUST send this parameter, and the server SHOULD reverse a long-term checkout of the file if all of the conditions are met to do so.

force: A **BOOLEAN** (section 2.2.2.1.3) that reverses the checkout of a file by another user. The server MAY ignore this value. If the server chooses to implement this functionality, it SHOULD do additional authorization checks and ignore the parameter if those checks fail. The server MUST default this argument to FALSE if not sent by the client.<53>

rlsshortterm: A **BOOLEAN** value that indicates if the client wants to release a short-term checkout or a long-term checkout. If TRUE, the server MUST release the short-term checkout lock; otherwise, the

server SHOULD release a long-term checkout that the client has acquired. The server SHOULD return an appropriate error if the client does not have the kind of checkout it is trying to undo.

time_checked_out: A **TIME** (section 2.2.2.1.6) that indicates the client's record of the time at which the file was last checked out. The server MAY refuse to revert a checkout if the time does not match the server's record of the time the file was checked out.

validateWelcomeNames: For semantics, see *validateWelcomeNames* in section 3.1.5.3.1.

Entry Point

FPAuthorScriptUrl

Return Values

meta_info: For semantics, see **meta_info** in section 3.1.5.3.1. This **METADICT** (section 2.2.2.2.11) contains all the metadata available to the client for the **document** that has been unchecked out after the method is complete.

3.1.5.3.41 url to web url

The **url to web url** request is used by the client to parse a URL into the **server-relative URL** of the **site** that contains the URL, and the service-relative URL for the file or **folder** within the site.

Parameters

service_name: This parameter is deprecated. See *service_name* in section 3.1.5.3.1.

url: A **URL-STRING** (section 2.2.2.2.3) that specifies the server-relative URL that the client wants to have parsed.

flags: Contains an **INT** (section 2.2.2.1.2) that MUST be ignored by the server but MAY be sent by the client and SHOULD equal 0.

Entry Point

FPShtmlScriptUrl

Return Values

webUrl: A **URL-STRING** that specifies the server-relative URL of the site.

fileUrl: A **URL-STRING** that specifies the service-relative URL of the file.

3.1.5.4 Higher-Layer Triggered Events

There are no higher-layer triggered events for the FrontPage Server Extensions: Website Management Protocol server. Each client request is triggered by the client application's needs.

3.1.6 Timer Events

3.1.6.1 Short-Term Checkout Timer Expiry

When the short-term checkout timer on a **document** expires, the server MUST clear the short-term checkout on the document. This leaves the document open for editing by any user. If the client wants to prevent short-term checkout from expiring, the client MUST send another checkout document request for the same document before the checkout has expired.

3.1.7 Other Local Events

There are no other local events.

4 Protocol Examples

The following sections describe several operations as used in common scenarios to illustrate the function of the FrontPage Server Extensions: Website Management Protocol.

4.1 Example Entry Point for FrontPage Server Extensions

4.1.1 First Determining the Entry Point

Each method specification gives an entry point that corresponds to one of four URLs that are returned when a client performs an HTTP GET on _vti_inf.html. This section details how to determine the URL to POST given the known entry point.

4.1.1.1 First Entry Point Example

If the client wants to call the **server version** (section 3.1.5.3.36) method, it needs to use the FPShtmlScriptUrl entry point. For details, see section 3.1.3. If it is making this call against the root of the server, the URL is as follows.

```
/_vti_bin/shtml.dll/_vti_rpc
```

If the client is making a call against a **subsite** located at /search/, the URL is as follows.

```
/search/_vti_bin/shtml.dll/_vti_rpc.
```

4.1.1.2 Second Entry Point Example

If the client wants to call the **open service** (section 3.1.5.3.24) method, it needs to use the FPAuthorScriptURL entry point.

```
POST
/site_url/_vti_bin/_vti_aut/author.dll HTTP/1.0
.
.
.
method=open+service:12.0.n.nnnn
```

The first line shows a post to /site_url/_vti_bin/_vti_aut/author.dll, which is the FPAuthorScriptURL entry point for the **subsite** called 'site_url'.

4.1.2 SharePoint Services Entry Note

The TPScriptUrl entry point is present only on servers that have Windows® SharePoint® Services (or SharePoint Team Services) enabled. It is a service-relative URL and refers to the URL to POST for Microsoft Windows® SharePoint Services methods. Windows® SharePoint Services methods are not discussed in this document.

4.2 Example Trace for Posts

The following is an example trace for common operations that are performed on the client. The example shows operations such as opening a web **folder**; copying and pasting to (or from) a web folder; opening a file; saving changes in a file; and closing a file.

In this example, the WWW-Authenticate headers (as specified in [RFC2616] section 14.47) have been removed. "DOMAIN1" is a placeholder for **domain name**, "testuser" is a placeholder for username, and "fpseserver" is a placeholder for an actual server name. All the lines, except for any text files that are uploaded, SHOULD be terminated by "\n" rather than "\r\n", which is standard on Windows operating systems.

4.2.1 Querying for URLs to POST

Any user action that requires the client to interact with the server through Microsoft FrontPage Server Extensions requires the client to know what URLs to POST. Consequently, any FrontPage Server Extensions: Website Management Protocol conversation will begin with the client posting an HTTP GET to /_vti_inf.html to determine the URLs of author.dll, shtml.dll (for details, see section 3.1.3.2.1).

4.2.1.1 Client HTTP GET Request for _vti_inf.html

```
GET /_vti_inf.html HTTP/1.1
Date: Thu, 08 June 2006 21:39:52 GMT
MIME-Version: 1.0
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MS FrontPage 12.0)
Host: fpseserver
Accept: auth/sicily
Content-Length: 0
Connection: Keep-Alive
Cache-Control: no-cache
```

4.2.1.2 Server HTTP Response

```
HTTP/1.1 200 OK
Content-Length: 1754
Content-Type: text/html
Last-Modified: Thu, 08 June 2006 21:04:13 GMT
Accept-Ranges: bytes
ETag: "2f7ad4cbe6fc61:33a"
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0 Pub
Date: Thu, 08 June 2006 21:39:42 GMT

<!-- FrontPage Configuration Information
FPVersion="12.0.0.000"
FPShtmlScriptUrl="_vti_bin/shtml.dll/_vti_rpc"
FPAuthorScriptUrl=" vti bin/ vti aut/author.dll"
FPAdminScriptUrl="_vti_bin/_vti_adm/admin.dll"
TPScriptUrl="_vti_bin/owssvr.dll"
-->
```

4.2.2 Opening a Web Folder

This example uses the **server version** (section 3.1.5.3.36) request to enumerate the **documents** in the root of the server. This part of the example corresponds to opening a **folder** as a web folder in a web browser.

4.2.2.1 Client Calls server version Method

```
Date: Thu, 08 June 2006 21:39:52 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 42
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=server+version%3a12%2e0%2e0%2e3417
```

4.2.2.2 Server Responds to server version Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:39:42 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=server version:5.0.2.6738
<p>server version=
<ul>
<li>major ver=5
<li>minor ver=0
<li>phase ver=2
<li>ver incr=6738
</ul>
<p>source control=1
</body>
</html>
```

4.2.2.3 Client Calls list documents Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:01 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 336
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=list+documents%3a5%2e0%2e2%2e6738&service%5fname=
&listHiddenDocs=false&listExplorerDocs=false&listRecurse=
false&listFiles=true&listFolders=true&listLinkInfo=
false&listIncludeParent=true&listDerived=false&listBorders=
false&listChildWebs=true&listThickets=true&initialUrl=&folderList=
%5b%3bTW%7c08+June+2006+21%3a04%3a14+%2d0000%5d
```

4.2.2.4 Server Responds to list documents Method

```
HTTP/1.1 200 OK
```

```
Connection: close
Date: Thu, 08 June 2006 21:39:51 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=list documents:5.0.2.6738
<p>document list=
<ul>
<ul>
<li>document_name=Thicket test.htm
<li>meta_info=
<ul>
<li>vti author
<li>SR|DOMAIN1&#92;testuser
<li>vti modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:28:52 -0000
<li>vti timecreated
<li>TR|08 June 2006 21:27:31 -0000
<li>vti_title
<li>SW|Test
<li>vti_nexttolasttimemodified
<li>TR|08 June 2006 21:28:39 -0000
<li>vti filesize
<>IR|930
<li>vti metatags
<li>VR|HTTP-EQUIV&#61;Content-Type text/html&#59;&#92;charset&#61;
windows-1252 Generator Microsoft&#92; Word&#92; 12&#92; (filtered)
<li>vti charset
<li>SR|windows-1252
<li>vti generator
<li>SR|Microsoft Word 12 (filtered)
<li>vti_timelastwritten
<li>TX|08 June 2006 21:28:52 -0000
</ul>
</ul>
</ul>
<p>urldirs=
<ul>
<ul>
<li>url=
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|true
<li>vti dirlateststamp
<li>TW|08 June 2006 21:28:52 -0000
</ulul>
</ul>
<ul>
<li>url=aspnet_client
<li>meta info=
<ul>
<li>vti isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
```

```
<li>BR|false
<li>vti_hassubdirs
<li>BR|true
</ul>
</ul>
<ul>
<li>url=images
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
<ul>
<li>url=Thicket Test_files
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
<ul>
<li>url=_private
<li>meta info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti isbrowsable
<li>BR|false
<li>vti isscriptable
<li>BR|false
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
</ul>
</body>
</html>
```

4.2.3 Copying a File to a Web Folder

This example uses the **url to web url** (section 3.1.5.3.41) request to discover where a file (in this case, /small.txt) belongs, and this example uses the **put document** (section 3.1.5.3.25) request to upload it. This part of the example corresponds to a copy/paste operation into the web **folder**.

4.2.3.1 Client Calls url to web url Method

```
POST /_vti_bin/shtml.dll/_vti_rpc HTTP/1.1
Date: Thu, 08 June 2006 21:40:17 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
```

```
Content-Length: 68
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=url+to+web+url%3a5%2e0%2e2%2e6738&url=%2fsmall%2etxt&flags=0
```

4.2.3.2 Server Responds to url to web url Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:07 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=url to web url:5.0.2.6738
<p>webUrl=/
<p>fileUrl=small.txt
</body>
</html>
```

4.2.3.3 Client Calls put document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:17 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 224
Content-Type: application/x-vermeer-urlencoded
X-Vermeer-Content-Type: application/x-vermeer-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=put+document%3a5%2e0%2e2%2e6738&service%5fname=&document=%5b
document%5fname%3dsmall%2etxt%3bmeta%5finfo%3d%5b%5d%5d&put%5foption
=edit%2catomic%2cthicket&comment=&keep%5fchecked%5fout=false
This is a small text file.
```

4.2.3.4 Server Responds to put document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:07 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0 Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=put document:5.0.2.6738
<p>message=successfully put document 'small.txt' as 'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
```

```
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti_backlinkinfo
<li>VX|
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
</ul>
</ul>
</body>
</html>
```

4.2.4 Downloading a File from a Web Folder

This example uses the **get document** (section 3.1.5.3.11) request to download the file. This part of the example corresponds to a copy/paste operation from the web **folder**.

4.2.4.1 Client Calls get document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:30 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 162
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=get+document%3a5%2e0%2e2%2e6738&service%5fname=&document
%5fname=small%2etxt&old%5ftheme%5fhtml=false&force=true&get
%5foption=none&doc%5fversion=&timeout=0
```

4.2.4.2 Server Responds to get document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:20 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=get document:5.0.2.6738
<p>message=successfully retrieved document 'small.txt' from
'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
```

```
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti backlinkinfo
<li>VX|
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
</ul>
</ul>
</body>
</html>
This is a small text file.
```

4.2.5 Opening a File in a Web Folder

When opening a file, as seen in the next part of the example, a client application calls the **get document** (section 3.1.5.3.11) request with a time-out of a 10-minute short-term checkout, as can be seen at the end of the **get document** request in the following section. This guarantees that the **document** cannot be modified by other users while it is open in the client application.

4.2.5.1 Client Calls get document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:45 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 175
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=get+document%3a5%2e0%2e2%2e6738&service%5fname=&document%5f
name=small%2etxt&old%5ftheme%5fhtml=false&force=false&get%5foption=
chkoutExclusive&doc%5fversion=&timeout=10
```

4.2.5.2 Server Responds to get document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:35 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=get document:5.0.2.6738
<p>message=successfully retrieved document 'small.txt' from
'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
```

```
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti_backlinkinfo
<li>VX|
<li>vti_sourcecontrollockexpires
<li>TR|08 June 2006 21:51:35 -0000
<li>vti_sourcecontrolcheckedoutby
<li>SR|DOMAIN1&#92;testuser
<li>vti_sourcecontrolmultiuserchkoutby
<li>VR|DOMAIN1&#92;&#92;testuser
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
</ul>
</ul>
</body>
</html>
This is a small text file.
```

4.2.6 Saving a File to a Web Folder

Changing and saving a file, as seen in the next part of the example, requires calling the **put document** (section 3.1.5.3.25) request.

4.2.6.1 Client Calls put document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:57 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 290
Content-Type: application/x-vermeer-urlencoded
X-Vermeer-Content-Type: application/x-vermeer-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=put+document%3a5%2e0%2e2%2e6738&service%5fname=&document=%5
bdocument%5fname%3dsmall%2etxt%3bmeta%5finfo%3d%5bvti%5ftimelastmodi
fied%3bTW%7c08+June+2006+21%3a40%3a07+%2d0000%5d%5d&put%5foption=edi
t&comment=&keep%5fchecked%5fout=false
This is a small text file. Now, a little bigger.
```

4.2.6.2 Server Responds to put document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:47 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc
```

```
<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=put document:5.0.2.6738
<p>message=successfully put document 'small.txt' as 'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:41:47 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_backlinkinfo
<li>VX|
<li>vti_sourcecontrollockexpires
<li>TR|08 June 2006 21:51:35 -0000
<li>vti_sourcecontrolcheckedoutby
<li>SR|DOMAIN1&#92;testuser
<li>vti_sourcecontrolmultiuserchkoutby
<li>VR|DOMAIN1&#92;&#92;testuser
<li>vti_nexttolasttimemodified
<li>TW|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|51
<li>vti_timelastwritten
<li>TX|08 June 2006 21:41:47 -0000
</ul>
</ul>
</body>
</html>
```

4.2.7 Closing a File

Finally, this example shows what happens when the file is closed in the client application, which requires a call to the **uncheckout document** (section 3.1.5.3.40) request to release the lock. Note that the example does not illustrate the effects of waiting 10 minutes to cause the client application to renew the short-term checkout, which would have caused a checkout document (section 3.1.5.3.7) request to be sent with a *timeout* parameter.

4.2.7.1 Calls uncheckout document Method

```
POST / vti bin/ vti aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:59 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 120
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=uncheckout+document%3a5%2e0%2e2%2e6738&service%5fname=
&document%5fname=small%2etxt&force=false&rlsshortterm=true
```

4.2.7.2 Server Responds to uncheckout document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:49 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-Type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=uncheckout document:5.0.2.6738
<p>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:41:47 -0000
<li>vti timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_backlinkinfo
<li>VX|
<li>vti_nexttolasttimemodified
<li>TW|08 June 2006 21:40:07 -0000
<li>vti filesize
<li>IR|51
<li>vti timelastwritten
<li>TX|08 June 2006 21:41:47 -0000
</ul>
</body>
</html>
```

5 Security

The following sections specify the security considerations for implementers.

5.1 Security Considerations for Implementers

5.1.1 One-Click Attacks

It is possible for an attacker to lure a user to a malicious **page**, for example by sending the user a URL in an email. When the user visits the malicious page, that page can perform a silent POST to the server. Because the FrontPage Server Extensions: Website Management Protocol is merely an HTTP POST, this means that the attacker can lure the user into performing any FrontPage Server Extensions: Website Management Protocol operation against any server. This sort of attack is termed a one-click attack.

To prevent this type of attack, servers SHOULD require all incoming FrontPage Server Extensions: Website Management Protocol requests to have the HTTP header X-Vermeer-Content-Type, as specified in [RFC2616] section 14.17. Because normal web browsers do not send this header, requiring it effectively prevents users from browsing to a page that can execute a silent FrontPage Server Extensions: Website Management Protocol method call. It is strongly recommended that all implementations of the FrontPage Server Extensions: Website Management Protocol require this header to prevent one-click attacks.

5.1.2 Permissions for Entry Points

Servers have traditionally restricted access to methods to certain classes of users. Although this restriction is not required by the FrontPage Server Extensions: Website Management Protocol, it is recommended because some methods, such as the **remove documents** (section 3.1.5.3.31) method, can be damaging to user data.

The FrontPage Server Extensions: Website Management Protocol has traditionally determined which users can call which methods based on the method entry points. Methods whose entry point is FPShtmlScriptUrl can usually be called by any user. Methods with the FPAuthorScriptURL entry point are restricted to users who can read or write documents on the server. The reason for this model is that methods such as **remove documents** are considered more dangerous than the **server version** (section 3.1.5.3.36) method. As such, restricting unauthenticated users from even calling the more powerful methods provides an extra layer of security.

Implementers of the FrontPage Server Extensions: Website Management Protocol are free to restrict method entry point security if they choose to, or they can rely on the object permissions discussed in the following section.

5.1.3 Permissions for Objects

Like most file systems, FrontPage Server Extensions: Website Management Protocol objects can have a notion of security. The granularity of this security is up to the server implementers. Microsoft Windows® implementations of the FrontPage Server Extensions: Website Management Protocol provide the capability to granularly control read access and write access on files, folders, and services. On a secured server, each method call SHOULD check the appropriate rights before executing. If the user does not have sufficient rights, the implementation SHOULD trigger the HTTP layer to return a 401 message, access denied. The HTTP layer on the client and server SHOULD then manage authenticating the user, if that user does in fact have permissions.

5.2 Index of Security Parameters

There are no security parameters in the FrontPage Server Extensions: Website Management Protocol.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Windows NT
- Windows 2000 Professional operating system
- Windows XP operating system
- Windows Server 2003 operating system
- Windows Vista operating system
- Windows Server 2008 operating system
- Windows Server 2008 R2 operating system

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 1.7.1: Specific protocol versions implemented by FPSE and SharePoint, covered in [MC-FPSEWM] and [MS-FPSE], are listed in the following table:

Protocol version	FPSE version	SharePoint version	Windows-To-Windows behaviors: described in MS-FPSE	Broader behaviors: described in MC-FPSEWM
1.0.x.x	Vermeer FrontPage 1.0	n/a	No	No
1.1.x.x	Microsoft FrontPage 1.1	n/a	No	No
2.0.x.x	Microsoft FrontPage 97	n/a	No	No
3.0.x.x	Microsoft FrontPage 98	n/a	No	No
4.0.x.x	Microsoft FrontPage 2000	n/a	Yes	No
5.0.x.x	Microsoft FrontPage 2002	Windows SharePoint Team Services 1.0	Yes	No

Protocol version	FPSE version	SharePoint version	Windows-To-Windows behaviors: described in MS-FPSE	Broader behaviors: described in MC-FPSEWM
6.0.x.x	n/a	Windows SharePoint Services 2.0	Yes	No
12.0.x.x	n/a	Windows SharePoint Services 3.0	Yes	Yes
14.0.x.x	n/a	Microsoft SharePoint Foundation 2010	Yes	Yes
15.0.x.x	n/a	Microsoft SharePoint Foundation 2013, Microsoft SharePoint Server 2016, Microsoft SharePoint Server 2019	n/a	n/a

<2> Section 2.2.2.2.5: If the version is at least SharePoint Foundation 2010, it does not implement the **DOC-INFO-REQUEST** type (section 2.2.2.2.15).

<3> Section 2.2.2.2.15: If the server version is at least SharePoint Foundation 2010, it does not implement this type.

<4> Section 2.2.2.2.18: The FrontPage Server Extensions: Website Management Protocol version 12.0 accepts this parameter.

<5> Section 2.2.2.2.18: The FrontPage Server Extensions: Website Management Protocol version 12.0 accepts this parameter.

<6> Section 2.2.2.2.18: The FrontPage Server Extensions: Website Management Protocol version 12.0 accepts this parameter and requires the requesting user to be a web administrator.

<7> Section 2.2.2.3.4: In FrontPage Server Extensions: Website Management Protocol Server version 12.0, this metakey contains the **server-relative URL** of the **service** with the default value "/_layouts/settings.aspx" appended to the end.

<8> Section 2.2.2.3.21: FrontPage Server Extensions: Website Management Protocol version 12.0 provides the following default list of categories when a site is created:

- Travel
- Expense Report
- Business
- Competition
- Goals/Objectives
- Ideas
- Miscellaneous
- Waiting
- VIP

- In Process
- Planning
- Schedule

<9> Section 2.2.2.3.28: FrontPage 2003 and SharePoint Designer 2007 use this metakey. Windows SharePoint Services does not use this metakey.

<10> Section 2.2.2.3.56: FrontPage Server Extensions: Website Management Protocol version 12.0 always sets this value to 1.

<11> Section 2.2.2.3.58: The Windows operating system client uses this metadata to avoid fetching the content of the file just to discover META tags with NAME="progid" and NAME="generator"; these are used to display icons for HTML files and to select an appropriate editor.

<12> Section 2.2.2.3.67: Windows SharePoint Services 3.0 does not return the **vti_setuppath** (section 2.2.2.3.67) parameter.

<13> Section 2.2.2.3.70: Due to a programming defect, if the version is at least 12.0 of the FrontPage Server Extensions: Website Management Protocol, the server does not return this metakey in the **checkin document** (section 3.1.5.3.6) or **put document** (section 3.1.5.3.25) methods. However, the server will return this metakey in the **getDocsMetaInfo** (section 3.1.5.3.16) and **list documents** (section 3.1.5.3.20) methods.

<14> Section 2.2.2.3.71: The FrontPage Server Extensions: Website Management Protocol does not have a method to set this metakey for a document. Windows SharePoint Services 3.0 sets this metakey in response to methods invoked through a SOAP-based protocol.

<15> Section 2.2.2.3.85: The default value for this metakey in FrontPage Server Extensions: Website Management Protocol version 12.0 servers is "/_layouts/1033/toolpane.aspx".

<16> Section 3.1.1.1: The FrontPage Server Extensions: Website Management Protocol version 12.0 does not allow users to turn the source control sandbox off.

<17> Section 3.1.2.1: All Windows operating system clients request a short-term checkout length of ten minutes. The clients attempt to renew the short-term checkout 10 seconds before it expires.

<18> Section 3.1.3.2.1: Windows Vista operating system does not perform this GET, and instead assumes the values shown in the example in section 3.1.3.2.1.

<19> Section 3.1.5.1: If the client does not include FrontPage in its User-Agent string, the Windows NT, Windows Server 2003 operating system, Windows Server 2008 operating system with Service Pack 2 (SP2), and Windows Server 2008 R2 operating system operating systems will respond with the HTTP Content-Type as "text/html" and present more simplistic error strings.

<20> Section 3.1.5.2: Version 12.0 of the FrontPage Server Extensions: Website Management Protocol server will treat unknown arguments as a syntax error if the method takes any parameters. For methods that take no parameters, such as **server version**, the FrontPage Server Extensions: Website Management Protocol server will ignore the parameters.

<21> Section 3.1.5.2: Due to a programming defect, FrontPage Server Extensions: Website Management Protocol server version 12.0 will erroneously return a badly formed response message body that is not compliant with [RFC2616] for most method calls made without authentication that result in an HTTP 401 error response.

The following is an example of this badly formed message body that is produced in this case:

```
<html dir="ltr">
  <HEAD>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
     name="CharsetDefinition">
  </HEAD>
  <body ID=idErr>
    <p><H2>Access denied.</H2></p>
    <p>You do not have permission to perform this action or access
       this resource.</p>
    <!-- commentElt Access denied. -->
  </body>
</html>
<html>
  <head>
    <title>vermeer RPC packet</title>
  </head>
  <body>
    <p>method=open service:12.0.0.4518
    <p>status=
    <ul>
      <li>status=917556
      <li>osstatus=0
      <li>msg=You are not authorized to execute this operation.
      <li>osmsg=
    </ul>
  </body>
</html>
```

Note The response message body created by the FrontPage Server Extensions server software that exhibit this defect is badly formed because of the presence of two separate <HTML> sections, which MAY cause unexpected behavior in an insufficiently robust client that attempts to render or otherwise make use of the body.

All existing FrontPage Server Extensions: Website Management Protocol clients ignore the message body, if any, returned with an HTTP 401 response. Because an update or future version of the FrontPage Server Extensions: Website Management Protocol server MAY correct this defect, clients MUST NOT rely on this defective server behavior.

<22> Section 3.1.5.3: The information for these requests applies to server extensions for versions of Microsoft FrontPage 2000, Microsoft FrontPage 2002, Microsoft FrontPage 2003, and Microsoft SharePoint Designer 2007.

<23> Section 3.1.5.3.1: FrontPage Server Extensions: Website Management Protocol version 12.0 servers require this parameter to have a value of at least 4.0.2.2611 if it is sent by a client.

<24> Section 3.1.5.3.1: FrontPage Server Extensions: Website Management Protocol version 12.0 sends this parameter.

<25> Section 3.1.5.3.1: The FrontPage Server Extensions: Website Management Protocol version 12.0 ignores this parameter.

<26> Section 3.1.5.3.1: The *service_name* parameter is sent by some Microsoft Office clients for some methods other than **create service** (section 3.1.5.3.8), **remove service** (section 3.1.5.3.32), and **rename service** (section 3.1.5.3.33), but this parameter is consistently ignored when not required by Windows SharePoint Services servers.

<27> Section 3.1.5.3.2: If the version is at least SharePoint Foundation 2010, it does not implement this method.

<28> Section 3.1.5.3.3: SharePoint Foundation 2010 does not implement this method.

<29> Section 3.1.5.3.10: The FrontPage Server Extensions: Website Management Protocol version 12.0 does not have the notion of an executable folder.

<30> Section 3.1.5.3.11: Windows SharePoint Services 3.0 does not support nonexclusive checkouts.

<31> Section 3.1.5.3.13: SharePoint Foundation 2010 does not implement this method.

<32> Section 3.1.5.3.14: SharePoint Foundation 2010 does not implement this method.

<33> Section 3.1.5.3.15: SharePoint Foundation 2010 does not implement this method.

<34> Section 3.1.5.3.17: SharePoint Foundation 2010 does not implement this method.

<35> Section 3.1.5.3.18: SharePoint Foundation 2010 does not implement this method.

<36> Section 3.1.5.3.19: SharePoint Foundation 2010 does not implement this method.

<37> Section 3.1.5.3.20: The FrontPage Server Extensions: Website Management Protocol clients send listDerived=false in the request and do not request the contents of a _derived folder.

<38> Section 3.1.5.3.20: The FrontPage Server Extensions: Website Management Protocol version 5.0 and version 12.0 servers return an empty **bot_list**.

<39> Section 3.1.5.3.21: SharePoint Foundation 2010 does not implement this method.

<40> Section 3.1.5.3.22: SharePoint Foundation 2010 does not implement this method.

<41> Section 3.1.5.3.26: The FrontPage Server Extensions: Website Management Protocol version 12.0 does not support this behavior.

<42> Section 3.1.5.3.27: SharePoint Foundation 2010 does not implement this method.

<43> Section 3.1.5.3.28: SharePoint Foundation 2010 does not implement this method.

<44> Section 3.1.5.3.29: SharePoint Foundation 2010 does not implement this method.

<45> Section 3.1.5.3.30: SharePoint Foundation 2010 does not implement this method.

<46> Section 3.1.5.3.31: The FrontPage Server Extensions: Website Management Protocol version 12.0 does not support this behavior.

<47> Section 3.1.5.3.31: The SharePoint Designer 2007 sends an empty **METADICT** (section 2.2.2.2.11).

<48> Section 3.1.5.3.31: The SharePoint Designer 2007 sends an empty **METADICT** (section 2.2.2.2.11).

<49> Section 3.1.5.3.32: The FrontPage Server Extensions: Website Management Protocol version 5.0 can delete a site even when there are subsites present.

<50> Section 3.1.5.3.35: SharePoint Foundation 2010 does not implement this method.

<51> Section 3.1.5.3.38: In Windows SharePoint Services 3.0, source control is always turned on and this method has no effect on the server.

<52> Section 3.1.5.3.38: SharePoint Foundation 2010 does not implement this method.

<53> Section 3.1.5.3.40: The FrontPage Server Extensions: Website Management Protocol version 12.0 requires that the user have a special break checkout right.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Description	Revision class
1.5 Prerequisites/Preconditions	Updated lowercase version of normative terms.	Minor
2.2.1 Syntax	Updated lowercase version of normative terms.	Minor
2.2.2.2.11 METADICT	Updated lowercase version of normative terms.	Minor
3.1.1 Abstract Data Model	Updated lowercase version of normative terms.	Minor
3.1.1.1 Source Control	Updated lowercase version of normative terms.	Minor
4.2 Example Trace for Posts	Updated lowercase version of normative terms.	Minor
5.1.1 One-Click Attacks	Updated lowercase version of normative terms.	Minor
5.1.3 Permissions for Objects	Updated lowercase version of normative terms.	Minor

8 Index