[MS-OXBBODY]:
Best Body Retrieval Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability.
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Revised and edited technical content.
09/03/2008	1.02		Revised and edited technical content.
10/01/2008	1.03		Revised and edited technical content.
12/03/2008	1.04		Updated IP notice.
03/04/2009	1.05		Revised and edited technical content.
04/10/2009	2.0		Updated technical content and applicable product releases.
07/15/2009	3.0.1	Minor	Revised and edited for technical content.
11/04/2009	3.1.1	Minor	Updated the technical content.
02/10/2010	3.1.1	None	Version 3.1.1 release
05/05/2010	3.1.2	Editorial	Revised and edited the technical content.
08/04/2010	3.1.2	No change	No changes to the meaning, language, or formatting of the technical content.
11/03/2010	3.1.2	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	3.2	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction 4
1.1 Glossary 4
1.2 References 4
1.2.1 Normative References 4
1.2.2 Informative References 5
1.3 Overview 5
1.4 Relationship to Protocols and Other Structures 5
1.5 Applicability Statement 5
1.6 Versioning and Localization 6
1.7 Vendor-Extensible Fields 6

2 Structures 7
2.1 Best Body Algorithm 7
2.2 Plain Text or HTML Converted to RTF 10
2.3 Special Considerations for S/MIME Secure Messages 10
2.4 Special Considerations for Rights-Managed Secure Messages 10
2.5 Obtaining the best body from the server 10

3 Structure Examples 11

4 Security Considerations 12

5 Appendix A: Product Behavior 13

6 Change Tracking 14

7 Index 16

1 Introduction

This document specifies the mechanism for determining the best format of storing **message bodies**.

In order to support clients that handle only one body format, servers can make any of the three body formats available on demand through alternate body properties. This document specifies how to determine which of the three body formats is the primary or "best" body format.

Exactly how message text is converted from one format to another, and to what extent formatting is preserved in the conversion, is implementation-dependent.

1.1 Glossary

The following terms are defined in [MS-OXGLOS]:

Hypertext Markup Language (HTML)
message body
Message object
plain text
recipient
Rich Text Format (RTF)
S/MIME (Secure/Multipurpose Internet Mail Extensions)

The following terms are specific to this document:

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

1.2 References

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-OXCDATA] Microsoft Corporation, "Data Structures", April 2008.

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol Specification", June 2008.

[MS-OXCROPS] Microsoft Corporation, "Remote Operations (ROP) List and Encoding Protocol Specification", June 2008.

[MS-OXORMMS] Microsoft Corporation, "Rights-Managed E-Mail Object Protocol Specification", June 2008.

[MS-OXOSMIME] Microsoft Corporation, "S/MIME E-Mail Object Protocol Specification", June 2008.

[MS-OXPROPS] Microsoft Corporation, "Exchange Server Protocols Master Property List", April 2008.

[MS-OXRTFEX] Microsoft Corporation, "Rich Text Format (RTF) Extensions Specification", June 2008.

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

1.2.2 Informative References

[MS-OXGLOS] Microsoft Corporation, "Exchange Server Protocols Master Glossary", April 2008.

1.3 Overview

Servers support three different formats for a message body:

- **Plain text** - The most accepted form of messaging format. Most e-mail message readers can display text messages in plain text format. However, plain text messages cannot display colors, different fonts, or emphasis such as bold or italic text.
- **Rich Text Format (RTF)** - RTF displays colors, fonts, and formatting.
- **HTML** - The message is sent as an HTML page, complete with tags to change the appearance of the text. The **recipient's** e-mail client program then formats and displays the HTML.

A client requests a specific message body format from a server by issuing a **RopGetPropertiesSpecific** ([MS-OXCROPS] section 2.2.8.3) or **RopOpenStream** ([MS-OXCROPS] section 2.2.9.1) request for the corresponding property. For more information, see [MS-OXCROPS]. The following table lists the properties that correspond to each body format, plus one important related property.

Body format	Property identifier	Property type	Description
Plain	**PidTagBody** ([MS-OXPROPS] section 2.683)	**PtypString**	Message body text in plain text format.
RTF	**PidTagRtfCompressed** ([MS-OXPROPS] section 2.1037)	**PtypBinary**	Message body text in RTF format.
HTML	**PidTagHtml** ([MS-OXPROPS] section 2.809)	**PtypBinary**	Message body text in HTML format.
Varies	**PidTagRtfInSync** ([MS-OXPROPS] section 2.1038)	**PtypBoolean**	Indicates whether **PidTagBody** and **PidTagRtfCompressed** contain the same text (ignoring formatting).

Because a server can provide the message body in any of these three formats, an algorithm to determine which is the "original", "primary", or "best" body is needed for clients that are capable of handling multiple body formats. For more details about that algorithm, see section 2.

1.4 Relationship to Protocols and Other Structures

This specification relies on [MS-OXCROPS], [MS-OXPROPS], and [MS-OXCDATA] for the specification of **RopGetPropertiesSpecific** ([MS-OXCROPS] section 2.2.8.3) request, property values, and status codes.

1.5 Applicability Statement

The algorithm in section 2.1 applies to **Message objects** of all types except the following:

- The value of **PidTagMessageClass** ([MS-OXPROPS] section 2.884) is exactly "IPM.Note.SMIME" (section 2.3). (**Note:** If the value of **PidTagMessageClass** is "IPM.Note.SMIME.MultipartSigned", then the algorithm in section 2.1 is applicable.)
- The value of **PidTagMessageClass** is "IPM.Note" and the value of **PidNameContentClass** ([MS-OXPROPS] section 2.432) is "rpmsg.Message" (section 2.4).

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

None.

2 Structures

2.1 Best Body Algorithm

Step 1. Issue a **RopGetPropertiesSpecific** request ([MS-OXCROPS] section 2.2.8.3) for five properties: **PidTagBody** ([MS-OXPROPS] section 2.683), **PidTagRtfCompressed** ([MS-OXPROPS] section 2.1037), **PidTagHtml** ([MS-OXPROPS] section 2.809), **PidTagRtfInSync** ([MS-OXPROPS] section 2.1038), and **PidTagNativeBody** ([MS-OXPROPS] section 2.902). For more details, see [MS-OXCMSG] and [MS-OXPROPS].

If the status code returned by **RopGetPropertiesSpecific** indicates a failure, then the body type is undefined and the algorithm exits.

If the client does not request all three of the body type properties (**PidTagBody**, **PidTagRtfCompressed**, and **PidTagHtml**), the server SHOULD return the best value for **PidTagNativeBody** that fits one of the requested body types.

If all five property values are retrieved and the **PidTagNativeBody** value is as specified in the following table, then the server has already saved the best body format to use in the **PidTagNativeBody** variable. It is not necessary to perform the algorithm below.

If the **PidTagNativeBody** property is returned, the following table corresponds to the particular body format. If this property is missing or if it is of any other value, proceed to the remaining steps of the algorithm.

PidTagNativeBody value	Property Identifier	Body format
1	**PidTagBody**	Plain
3	**PidTagRtfCompressed**	RTF
3	**PidTagHtml**	HTML

If **PidTagNativeBody** is not returned but the remaining four property values were retrieved, then the **RopGetPropertiesSpecific** operation returns a **StandardPropertyRow**, as specified in [MS-OXCDATA]. If any of the four property values were not retrieved, then the **RopGetPropertiesSpecific** operation returns a **FlaggedPropertyRow**, as specified in [MS-OXCDATA].

Step 2. Create four variables: *PlainStatus*, *RtfStatus*, *HtmlStatus*, and *RtfInSync*. Examine the returned property values and assign values to the corresponding variables as follows. In each case, if there is an error code, then it is either NotFound or NotEnoughMemory.

- *PlainStatus* – If **RopGetPropertiesSpecific** returned a **StandardPropertyRow**, or the **PidTagBody** value is **PtypString**, then assign the NoError error code to **PlainStatus**; else copy the error code from the **FlaggedPropertyValue** to *PlainStatus*.
- *RtfStatus* - If **RopGetPropertiesSpecific** returned a **StandardPropertyRow**, or the **PidTagRtfCompressed** value is **PtypBinary**, then assign NoError to **RtfStatus**; else copy the error code from the **FlaggedPropertyValue** to *RtfStatus*.
- *HtmlStatus* - If **RopGetPropertiesSpecific** returned a **StandardPropertyRow**, or the **PidTagHtml** value is **PtypBinary**, then assign NoError to **HtmlStatus**; else copy the error code from the **FlaggedPropertyValue** to *HtmlStatus*.

- *RtfInSync* - If **RopGetPropertiesSpecific** returned a **StandardPropertyRow**, or the **PidTagRtfInSync** value is **PtypBoolean**, then copy the **PtypBoolean** value to **RtfInSync**; else assign FALSE to *RtfInSync*.

Step 3. Determine the body format based on values of the four variables created in step 2. The following table can be implemented as an "if-then-else" chain, in exactly the order specified.

	PlainStatus	RtfStatus	HtmlStatus	RtfInSync	Body format
1	NotFound	NotFound	NotFound	Any	Undefined
2	NotEnoughMemory	NotFound	NotFound	Any	Plain text
3	NotEnoughMemory	NotEnoughMemory	NotFound	Any	RTF
4	NotEnoughMemory	NotEnoughMemory	NotEnoughMemory	True	RTF
5	NotEnoughMemory	NotEnoughMemory	NotEnoughMemory	False	HTML
6	Any	NoError or NotEnoughMemory	NoError or NotEnoughMemory	True	RTF
7	Any	NoError or NotEnoughMemory	NoError or NotEnoughMemory	False	HTML
8	NoError or NotEnoughMemory	NoError or NotEnoughMemory	Any	True	RTF
9.1	NoError or NotEnoughMemory	NoError or NotEnoughMemory	Any	False	Plain text
9.2	NotFound	NoError or NotEnoughMemory	NotFound	Any	RTF
9.3	NoError or NotEnoughMemory	NotFound	NotFound	Any	Plain text
9.4	NotFound	NotFound	NoError or NotEnoughMemory	Any	HTML
10	If no other case fits				Plain text

This table can be implemented in the following pseudocode. Each row of the table is one clause of an "if-else-if" chain. Within a row, each column is ANDed together to form the condition of an "if" clause. If there is a case that is not defined, then the BodyFormat is plain text.

	Code to implement

	`If PidTagNativeBody <> NotFound Then`
	`  BodyFormat = PidTagNativeBody`
	`Else`

1	`If ((PlainStatus = NotFound) And`
	`  (RtfStatus = NotFound) And`
	`  (HtmlStatus = NotFound)) Then`
	`  BodyFormat = Undefined`

	Code to implement

2	`ElseIf ((PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NotFound) And`
	`(HtmlStatus = NotFound)) Then`
	`BodyFormat = Plain`

3	`ElseIf ((PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NotFound)) Then`
	`BodyFormat = Rtf`

4	`ElseIf ((PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NotEnoughMemory) And`
	`(RtfInSync = True)) Then`
	`BodyFormat = Rtf`

5	`ElseIf ((PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NotEnoughMemory) And`
	`(RtfInSync = False)) Then`
	`BodyFormat = Html`

6	`ElseIf ((RtfStatus = NoError or RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NoError or HtmlStatus = NotEnoughMemory) And`
	`(RtfInSync = True)) Then`
	`BodyFormat = Rtf`

7	`ElseIf ((RtfStatus = NoError or RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NoError or HtmlStatus = NotEnoughMemory) And`
	`(RtfInSync = False)) Then`
	`BodyFormat = Html`

8	`ElseIf ((PlainStatus = NoError or PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NoError or RtfStatus = NotEnoughMemory) And`
	`(RtfInSync = True)) Then`
	`BodyFormat = Rtf`

9.1	`ElseIf ((PlainStatus = NoError or PlainStatus = NotEnoughMemory) And`
	`(RtfStatus = NoError or RtfStatus = NotEnoughMemory) And`
	`(RtfInSync = False)) Then`
	`BodyFormat = Plain`

9.2	`ElseIf ((PlainStatus = NotFound) And`
	`(RtfStatus = NoError or RtfStatus = NotEnoughMemory) And`
	`(HtmlStatus = NotFound) Then`
	`BodyFormat = Rtf`

9.3	`ElseIf ((PlainStatus = NoError or PlainStatus = NotEnoughMemory) And` `(RtfStatus = NotFound) And`

	Code to implement
	(HtmlStatus = NotFound) Then BodyFormat = Plain
9.4	ElseIf ((PlainStatus = NotFound) And (RtfStatus = NotFound) And (HtmlStatus = NoError or HtmlStatus = NotEnoughMemory) Then BodyFormat = Html
10	Else BodyFormat = Plain
	End If End If

2.2 Plain Text or HTML Converted to RTF

When the result of the algorithm in section 2.1 is RTF, it is possible to determine whether the RTF was generated from original plain text or HTML, as specified in [MS-OXRTFEX].

2.3 Special Considerations for S/MIME Secure Messages

The algorithm of section 2.1 yields an accurate result for a clear-signed **S/MIME** message (the value of **PidTagMessageClass** ([MS-OXPROPS] section 2.884) is "IPM.Note.SMIME.MultipartSigned"). Its result is undefined for other types of S/MIME messages (the value of **PidTagMessageClass** is "IPM.Note.SMIME"). For a detailed specification of these message types, see [MS-OXOSMIME].

2.4 Special Considerations for Rights-Managed Secure Messages

For rights-managed secure messages, the message body properties specified in this document do not contain the actual message body; instead, they contain boilerplate text intended for recipients whose clients do not support rights-managed secure messages. The actual message body resides in an attachment and is not accessible as a property of the Message object. To obtain the actual message body, a client MUST decrypt and parse the attachment, as specified in [MS-OXORMMS].

While the algorithm in section 2.1 yields a result for rights-managed secure messages, that result applies to the boilerplate text and not to the actual message body.

2.5 Obtaining the best body from the server

If a message has already been saved to the server, then the client SHOULD<1> download the five properties and check the **PidTagNativeBody** property ([MS-OXPROPS] section 2.902) first. If the **PidTagNativeBody** property is set, then it is not necessary to perform the algorithm specified in section 2.1 to determine the best possible body.

3 Structure Examples

In the following example, a simple HTML message is sent to a server.

```
From: <user1@example.com>
To: <user2@example.com>
Subject: test HTML message
Date: Tue, 24 Jan 2006 01:58:57 -0800
MIME-Version: 1.0
Content-Type: text/html
Content-Transfer-Encoding: 7bit
Content-Class: urn:content-classes:message
Importance: normal

<HTML><BODY>Test message, <b>please</b> delete.</BODY></HTML>
```

The four property values of interest appear as follows.

Property	Value
PidTagBody ([MS-OXPROPS] section 2.683)	error, NotEnoughMemory
PidTagHtml ([MS-OXPROPS] section 2.809)	<HTML><HEAD><meta HTTP-equiv="Content-Type" content="text/HTML; charset=iso-8859-1"></HEAD><BODY>Test message, please delete.</BODY></HTML>
PidTagRtfCompressed ([MS-OXPROPS] section 2.1037)	error, NotEnoughMemory
PidTagRtfInSync ([MS-OXPROPS] section 2.1038)	FALSE

The algorithm of section 2.1 creates the four variables shown in the following table.

Property	Value
PlainStatus	NotEnoughMemory
RtfStatus	NotEnoughMemory
HtmlStatus	NoError
RtfInSync	FALSE

The algorithm in section 2.1 uses the four newly created variables and matches clause 7.

	Code to implement
7	`else if ((RtfStatus = NoError or RtfStatus = NotEnoughMemory) and (HtmlStatus = NoError or HtmlStatus = NotEnoughMemory) and (RtfInSync = False)) then`

And the result returned is HTML body format.

4 Security Considerations

For more information, refer to sections 2.3 and 2.4.

5 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® Exchange Server 2003
- Microsoft® Exchange Server 2007
- Microsoft® Exchange Server 2010
- Microsoft® Office Outlook® 2003
- Microsoft® Office Outlook® 2007
- Microsoft® Outlook® 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product 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 2.5: Not supported by Exchange 2003 or Exchange 2007.

6 Change Tracking

This section identifies changes that were made to the [MS-OXBBODY] protocol document between the November 2010 and March 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

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 or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.1 Best Body Algorithm	Added requirement for server to return the best value for PidTagNativeBody that matches the requested body type properties.	N	New content added.

7 Index

A

Applicability 5

C

Change tracking 14

E

Examples 11

F

Fields - vendor-extensible 6

G

Glossary 4

I

Implementer - security considerations 12
Informative references 5
Introduction 4

L

Localization 6

N

Normative references 4

O

Overview 5

P

Product behavior 13

R

References
  informative 5
  normative 4
Relationship to protocols and other structures 5

S

Security - implementer considerations 12

T

Tracking changes 14

V

Vendor-extensible fields 6
Versioning 6