[MS-OXBBODY]:

Best Body Retrieval Algorithm

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.
- **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.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial Availability.
4/25/2008	0.2	Minor	Revised and updated property names and other technical content.
6/27/2008	1.0	Major	Initial Release.
8/6/2008	1.01	Minor	Revised and edited technical content.
9/3/2008	1.02	Minor	Revised and edited technical content.
10/1/2008	1.03	Minor	Revised and edited technical content.
12/3/2008	1.04	Minor	Updated IP notice.
3/4/2009	1.05	Minor	Revised and edited technical content.
4/10/2009	2.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.1	Minor	Revised and edited for technical content.
11/4/2009	3.1.1	Minor	Updated the technical content.
2/10/2010	3.1.1	None	Version 3.1.1 release
5/5/2010	3.1.2	Editorial	Revised and edited the technical content.
8/4/2010	3.2	Minor	Clarified the meaning of the technical content.
11/3/2010	3.2	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	3.3	Minor	Clarified the meaning of the technical content.
8/5/2011	4.0	Major	Significantly changed the technical content.
10/7/2011	4.0	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	5.0	Major	Significantly changed the technical content.
4/27/2012	5.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	5.1	Minor	Clarified the meaning of the technical content.
10/8/2012	5.2	Minor	Clarified the meaning of the technical content.
2/11/2013	5.2	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	5.2	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	5.3	Minor	Clarified the meaning of the technical content.
2/10/2014	5.3	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	5.3	None	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
7/31/2014	5.3	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	5.3	None	No changes to the meaning, language, or formatting of the technical content.
3/16/2015	6.0	Major	Significantly changed the technical content.
5/26/2015	6.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2015	6.0	None	No changes to the meaning, language, or formatting of the technical content.
6/13/2016	6.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The Best Body Retrieval Algorithm determines the best format of a **message body**. This algorithm enables clients to determine the format of the message body that is most like the original message, and maintains the richness of the text and formatting in the original message.

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

Sections 1.6 and 2 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

best body: The text format that provides the richest representation of a **message body**. The algorithm for determining the best-body format is described in [MS-OXBBODY].

Hypertext Markup Language (HTML): An application of the Standard Generalized Markup Language (SGML) that uses tags to mark elements in a document, as described in [HTML].

message body: The main message text of an email message. A few properties of a **Message object** represent its message body, with one property containing the text itself and others defining its code page and its relationship to alternative body formats.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

remote operation (ROP): An operation that is invoked against a server. Each ROP represents an action, such as delete, send, or query. A ROP is contained in a ROP buffer for transmission over the wire.

Rich Text Format (RTF): Text with formatting as described in [MSFT-RTF].

rights-managed email message: An email message that specifies permissions that are designed to protect its content from inappropriate access, use, and distribution.

ROP request: See ROP request buffer.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [RFC5751].

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-OXCDATA] Microsoft Corporation, "Data Structures".

[MS-OXCMAPIHTTP] Microsoft Corporation, "Messaging Application Programming Interface (MAPI) Extensions for HTTP".

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

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

[MS-OXCRPC] Microsoft Corporation, "Wire Format Protocol".

[MS-OXORMMS] Microsoft Corporation, "Rights-Managed Email Object Protocol".

[MS-OXOSMIME] Microsoft Corporation, "S/MIME Email Object Algorithm".

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

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

[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

1.2.2 Informative References

None.

1.3 Overview

The **best body** algorithm determines the original, primary, or best format in which to transmit a message body to a client. The best body algorithm enables clients that are capable of handling multiple message body formats to determine which of the formats that they support is most like the original message. Requesting a message by using the best body algorithm maintains as much of the richness of the text and formatting in the original message as possible. The algorithm uses a combination of **remote operations (ROPs)** and property values to determine the best body format of the message. The best body format can be one of the following formats:

- **Plain text** – This format cannot display colors, different fonts, or emphasis such as bold or italic text. Plain text is the most accepted messaging format. Most e-mail message readers can display messages in plain text format.
- **Rich Text Format (RTF)** – This format displays colors, different fonts, emphasis, and formatting, such as bullets, text alignment, and linked objects.
- **HTML** – This format 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.

1.4 Relationship to Protocols and Other Algorithms

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

For conceptual background information and overviews of the relationships and interactions between this and other protocols, see [MS-OXPROTO].

1.5 Applicability Statement

The algorithm described in this document is used by a client to determine the format in which to retrieve a message from the server, when the client accepts multiple message body formats.

The best body algorithm applies to **Message objects** of all types except when the following conditions are true:

- The value of the **PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3) is exactly "IPM.Note.SMIME" (section 2.1.3.3). If the value of the **PidTagMessageClass** property is "IPM.Note.SMIME.MultipartSigned", then the algorithm described in section 2.1.3.1 is applicable.
- The value of the **PidTagMessageClass** property is "IPM.Note" and the value of the **PidNameContentClass** property ([MS-OXCMSG] section 2.2.1.48) is "rpmsg.Message", as described in section 2.1.3.4.

1.6 Standards Assignments

None.

2 Algorithm Details

2.1 Best Body Determination Algorithm Details

This section specifies the algorithm that determines the format of the message body that is most like the original message, in order to maintain the richness of the text and formatting in the original message.

2.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this algorithm. The described organization is provided to facilitate the explanation of how the algorithm behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

This protocol includes the following elements:

Global.Handle, as specified in [MS-OXCRPC] section 3.1.1.1.

session context cookie<1>, as specified in [MS-OXCMAPIHTTP] section 3.2.1.

MessageObject, as specified in [MS-OXCMSG] section 3.1.1.3. Additional elements for the **MessageObject** ADM type are defined in section 2.1.1.1.

2.1.1.1 Per Message Object

A Message object is represented by the **MessageObject** ADM data type. The following ADM element is maintained by the client for each **MessageObject** ADM type:

MessageObject.BestBody: The original, primary, or best message body format of the Message object.

2.1.2 Initialization

None.

2.1.3 Processing Rules

The processing rules for the best body algorithm as well as considerations for applying this algorithm to different message types are specified in sections 2.1.3.1 through section 2.1.3.4.

2.1.3.1 Best Body Algorithm

The Best Body Algorithm specifies the algorithm that the client uses to determine the best body format of a message.

Step 1. Issue a **RopGetPropertiesSpecific** ROP request ([MS-OXCROPS] section 2.2.8.3) for the following properties: **PidTagBody** ([MS-OXPROPS] section 2.609), **PidTagRtfCompressed** ([MS-OXPROPS] section 2.932), **PidTagHtml** ([MS-OXPROPS] section 2.724), and **PidTagRtfInSync** ([MS-OXPROPS] section 2.933). The client SHOULD also request **PidTagNativeBody** ([MS-OXPROPS] section 2.796).

If the **RopGetPropertiesSpecific** ROP returns a status code that 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<2> return the best value for the **PidTagNativeBody** property that fits one of the requested body types.

If the client retrieves all five property values and the value of the **PidTagNativeBody** property is as specified in the following table, then the server has already saved the best body format to use in the value of the **PidTagNativeBody** property. In this case, it is not necessary to perform the remainder of the algorithm specified in this section.

If the server returns a value for the **PidTagNativeBody** property, the client SHOULD use this value to determine the best body. Otherwise, the client proceeds to step 2. The following table identifies the best body format that corresponds to the value. If the **PidTagNativeBody** property is missing or the value is not provided in the following table, proceed to the remaining steps of the algorithm.

Property value	Property identifier	Body format
1	**PidTagBody**	Plain
2	**PidTagRtfCompressed**	RTF
3	**PidTagHtml**	HTML

If the server does not return the **PidTagNativeBody** property but does return the remaining four property values, then the **RopGetPropertiesSpecific** ROP returns a **StandardPropertyRow** structure ([MS-OXCDATA] section 2.8.1.1). If any of the four property values were not retrieved, then the **RopGetPropertiesSpecific** ROP returns a **FlaggedPropertyRow** structure ([MS-OXCDATA] section 2.8.1.2).

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 the value for the variable is either NotFound or NotEnoughMemory.

- *PlainStatus* – If the **RopGetPropertiesSpecific** ROP returned a **StandardPropertyRow** structure, or the value of the **PidTagBody** property is a **PtypString** ([MS-OXCDATA] section 2.11.1), then assign the NoError error code to the *PlainStatus* variable; else copy the error code from the **FlaggedPropertyValue** structure to the *PlainStatus* variable.
- *RtfStatus* - If the **RopGetPropertiesSpecific** ROP returned a **StandardPropertyRow** structure, or the value of the **PidTagRtfCompressed** property is a **PtypBinary** ([MS-OXCDATA] section 2.11.1) value, then assign the NoError error code to the *RtfStatus* variable; else copy the error code from the **FlaggedPropertyValue** structure to the *RtfStatus* variable.
- *HtmlStatus* - If the **RopGetPropertiesSpecific** ROP returned a **StandardPropertyRow** structure, or the value of the **PidTagHtml** property is a **PtypBinary** value, then assign the NoError error code to the *HtmlStatus* variable; else copy the error code from the **FlaggedPropertyValue** structure to the *HtmlStatus* variable.
- *RtfInSync* - If the **RopGetPropertiesSpecific** ROP returned a **StandardPropertyRow** structure, or the value of the **PidTagRtfInSync** property is **PtypBoolean** ([MS-OXCDATA] section 2.11.1), then copy the **PtypBoolean** value to the *RtfInSync* variable; else assign FALSE to the *RtfInSync* variable.

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

	PlainStatus	RtfStatus	HtmlStatus	RtfInSync	Body format
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 by using 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`

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

3	`ElseIf ((PlainStatus = NotEnoughMemory) And`

	Code to implement
	(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)

	Code to implement
	And (RtfStatus = NotFound) And (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

2.1.3.2 Determining Whether Plain Text or HTML Was Converted to RTF

When the result of the best body algorithm is RTF, as specified in section 2.1.3.1, the message body is parsed and reveals whether the RTF was generated from original plain text or HTML, as specified in [MS-OXRTFEX].

2.1.3.3 Special Considerations for S/MIME Secure Messages

The best body algorithm, as specified in section 2.1.3.1, yields an accurate result for a clear-signed **S/MIME (Secure/Multipurpose Internet Mail Extensions)** message, meaning the value of the **PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3) is "IPM.Note.SMIME.MultipartSigned". However, the result of the best body algorithm is undefined for other types of S/MIME messages, for example, when the value of the **PidTagMessageClass** property is "IPM.Note.SMIME". For details about these message types, see [MS-OXOSMIME].

2.1.3.4 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 best body algorithm, as specified in section 2.1.3.1, yields a result for rights-managed secure messages, that result applies to the boilerplate text and not to the actual message body.

A **PidTagMessageClass** property value of "IPM.Note" denotes a standard Message object, as specified in [MS-OXCMSG] section 2.2.1.3. A **PidNameContentClass** property value of "rpmsg.message" denotes a **rights-managed e-mail message**, as specified in [MS-OXORMMS] section 2.2.2.1.

3 Algorithm 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 are returned from the server with the following values.

Property name	Value
PidTagBody ([MS-OXPROPS] section 2.609)	error, NotEnoughMemory
PidTagHtml ([MS-OXPROPS] section 2.724)	<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 2.932)	error, NotEnoughMemory
PidTagRtfInSync ([MS-OXPROPS] section 2.933)	FALSE

The best body algorithm, as specified in section 2.1.3.1, creates the four variables shown in the following table.

Variable	Value
PlainStatus	NotEnoughMemory
RtfStatus	NotEnoughMemory
HtmlStatus	NoError
RtfInSync	FALSE

The best body algorithm uses the four newly created variables and matches clause 7, as specified in section 2.1.3.1.

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

And the result returned is HTML body format.

4 Security

4.1 Security Considerations for Implementers

None.

4.2 Index of Security Parameters

None.

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 Exchange Server 2013
- Microsoft Exchange Server 2016
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016

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 2.1.1: Exchange 2003, Exchange 2007, Exchange 2010, and the initial release of Exchange 2013 do not support the session context cookie. The session context cookie was introduced in Microsoft Exchange Server 2013 Service Pack 1 (SP1).

<2> Section 2.1.3.1: The **PidTagNativeBody** property ([MS-OXPROPS] section 2.796) is not supported by Exchange 2003 or Exchange 2007.

6 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

7 Index