



MAILMAN

DEVELOPER'S GUIDE

Version 8.0
August 2002
Revised September 2006

Department of Veterans Affairs
VistA Health Systems Design & Development (HSD&D)
Infrastructure and Security Services (ISS)

Revision History

Documentation Revisions

The following table displays the revision history for this document. Revisions to the documentation are based on patches and new versions released to the field.

Table i. Documentation revision history

Date	Revision	Description	Author
07/23/02	1.0	Initial MailMan V. 8.0 software and documentation release. MailMan V. 8.0 was first released as "DNS-Aware MailMan" in a supplemental document released in August 2002. However, the remaining MailMan documentation set was never updated.	Thom Blom and Gary Beuschel Oakland, CA Office of Information Field Office (OIFO)

Date	Revision	Description	Author
09/21/06	2.0	<p>MailMan V. 8.0 documentation reformatting/revision.</p> <p>Reformatted document to follow the latest ISS styles and guidelines.</p> <p>As of this date, all content updates have been completed for all released MailMan patches.</p> <p>Also, reviewed document and edited for the "Data Scrubbing" and the "PDF 508 Compliance" projects.</p> <p>Data Scrubbing—Changed all patient/user TEST data to conform to HSD&D standards and conventions as indicated below:</p> <p>The first three digits (prefix) of any Social Security Numbers (SSN) start with "000" or "666."</p> <p>Patient or user names are formatted as follows: XMPATIENT,[N] or XMUSER,[N] respectively, where the N is a number written out and incremented with each new entry (e.g., XMPATIENT, ONE, XMPATIENT, TWO, etc.).</p> <p>Other personal demographic-related data (e.g., addresses, phones, IP addresses, etc.) were also changed to be generic.</p> <p>PDF 508 Compliance—The final PDF document was recreated and now supports the minimum requirements to be 508 compliant (i.e., accessibility tags, language selection, alternate text for all images/icons, fully functional Web links, successfully passed Adobe Acrobat Quick Check).</p>	<p>MailMan Development Team Oakland, CA Office of Information Field Office (OIFO): Maintenance Project Manager— Jack Schram Project Planner—Laura Rowland Developer—Gary Beuschel Technical Writer—Thom Blom</p>

Patch Revisions

For a complete list of patches released with this software, please refer to the Patch Module on FORUM.

Contents

1.0	Introduction.....	1-1
1.1	Common Variables	1-2
1.2	Errors.....	1-3
2.0	Creating/Sending/Forwarding Messages	2-1
2.1	^XMA11	2-1
2.2	\$\$INFO^XMA11(): Edit "Information Only" Field.....	2-1
2.3	^XMA11A.....	2-1
2.3.1	WRITE^XMA11A: Send a Message (Interactive)	2-1
2.4	^XMA2	2-2
2.4.1	GET^XMA2: Create Message Stub	2-2
2.4.2	XMZ^XMA2: Create Message Stub	2-3
2.4.3	^XMD: Create and Send a Message	2-4
2.4.4	EN1^XMD: Add Text to a Message	2-5
2.4.5	ENL^XMD: Add Text to a Message	2-6
2.4.6	ENT^XMD: Send a Message (Interactive)	2-7
2.4.7	ENT1^XMD: Forward a Message (Address Restrictions Waived).....	2-7
2.4.8	ENT2^XMD: Forward a Message	2-8
2.5	^XMGAPI0	2-9
2.5.1	\$\$SUBCHK^XMGAPI0(): Validate Message Subject	2-9
2.6	^XMPG	2-10
2.6.1	ENT^XMPG: Create/Send PackMan Message with Globals ..	2-10
3.0	Editing Messages	3-1
3.1	^MXEDIT	3-1
3.2	CLOSED^MXEDIT(): "Close" Flag Toggle	3-1
3.2.1	CONFID^MXEDIT(): "Confidential" Flag Toggle.....	3-2
3.2.2	CONFIRM^MXEDIT(): "Confirm Receipt Requested" Flag Toggle 3-2	
3.2.3	DELIVER^MXEDIT(): Set/Delete Message Delivery Basket (All Users) 3-2	
3.2.4	INFO^MXEDIT(): "Information Only" Flag Toggle	3-3
3.2.5	PRIORITY^MXEDIT(): "Priority" Flag Toggle	3-3
3.2.6	SUBJ^MXEDIT(): Change Message Subject	3-3
3.2.7	TEXT^MXEDIT(): Replace Message Text.....	3-4
3.2.8	VAPOR^MXEDIT(): Set/Delete Message Vaporize Date	3-4
4.0	Message Actions	4-1
4.1	Parameter Definitions	4-1
4.2	^MXAPI	4-5
4.3	Message Actions—Building Block APIs.....	4-5
4.3.1	ADDRNSND^MXAPI(): Address and Send Message.....	4-5
4.3.2	CRE8XMZ^MXAPI(): Create a New Message Stub	4-5
4.3.3	TOWHOM^MXAPI(): Check One Message Addressee.....	4-6

4.3.4	VSUBJ^XMXAPI(): Validate a Subject.....	4-6
4.4	Message Actions—APIs	4-7
4.4.1	ANSRMSG^XMXAPI(): Answer a Message	4-7
4.4.2	DELMSG^XMXAPI(): Delete Messages from a Basket	4-8
4.4.3	FLTRMSG^XMXAPI(): Filter Messages in a Basket.....	4-8
4.4.4	FWDMSG^XMXAPI(): Forward Messages from a Basket	4-8
4.4.5	LATERMSG^XMXAPI(): "Later" Messages in a Basket.....	4-9
4.4.6	MOVEMSG^XMXAPI(): Move Messages to Another Basket.	4-10
4.4.7	PRTMSG^XMXAPI(): Print Messages	4-10
4.4.8	PUTSERV^XMXAPI(): Put a Message in a Server Basket....	4-11
4.4.9	REPLYMSG^XMXAPI(): Reply to Message	4-11
4.4.10	SEENBULL^XMXAPI(): Send a Bulletin	4-12
4.4.11	SENDMSG^XMXAPI(): Send a Message	4-13
4.4.12	TASKBULL^XMXAPI(): Send a Bulletin.....	4-13
4.4.13	TERMMMSG^XMXAPI(): Terminate Messages	4-14
4.4.14	VAPORMSG^XMXAPI(): Set Vaporize Date	4-14
4.4.15	ZAPSERV^XMXAPI(): Delete a Message from a Server Basket	4-15
5.0	Getting Information About and Text From Messages	5-1
5.1	^XMAH.....	5-1
5.1.1	ENT8^XMAH: Display a List of All Responses to a Message (Interactive).....	5-1
5.2	^XMGAPI0	5-1
5.2.1	\$\$SUBGET^XMGAPI0(): Get Message Subject.....	5-1
5.2.2	^XMGAPI1	5-2
5.2.3	\$\$READ^XMGAPI1(): Get a Line of Text from a Message.....	5-2
5.3	^XMGAPI2	5-2
5.3.1	\$\$HDR^XMGAPI2(): Set up an Array Containing Message Information 5-2	
5.3.2	^XML	5-5
5.3.2.1	GET^XML: Retrieve Next Line of Message Text	5-5
5.4	^XMRENT	5-6
5.4.1	\$\$NET^XMRENT(): Get Message Information	5-6
5.5	^XMS3	5-7
5.5.1	REC^XMS3: Get a Line of Text from a Message	5-7
6.0	Replies/Answers to Messages—Creating and Sending.....	6-1
6.1	^XMA11A.....	6-1
6.1.1	WRITE^XMA11A: Answer a Message (Interactive).....	6-1
6.2	^XMA2R.....	6-1
6.2.1	\$\$ENT^XMA2R(): Create/Send Reply and Get Message Number	6-1
6.2.2	\$\$ENTA^XMA2R(): Create/Send Answer and Get Message Number 6-2	
6.3	^XMAH1.....	6-3
6.3.1	^XMAH1: Create/Send a Reply to a Message (Interactive)	6-3
6.3.1.1	ENTA^XMAH1: Create/Send a Reply to a Message (Interactive)	6-4
7.0	Basket Actions.....	7-1

7.1	^XMA03	7-1
7.1.1	\$\$REN^XMA03: Perform Integrity Check on User Basket.....	7-1
7.2	^XMAD2.....	7-1
7.2.1	\$\$BSKT^XMAD2: Basket Lookup	7-1
7.3	^XMXAIB	7-2
7.3.1	CRE8BSKT^XMXAIB(): Create a Basket	7-2
7.3.2	CRE8MBOX^XMXAIB(): Create a Mailbox	7-2
7.3.3	DELBSKT^XMXAIB(): Delete a User's Basket	7-3
7.3.4	FLTRBSKT^XMXAIB(): Filter Messages in a Basket	7-3
7.3.5	FLTRMBOX^XMXAIB(): Filter All Messages in a User's Mailbox.....	7-4
7.3.6	LISTBSKT^XMXAIB(): Get a List of Baskets in a Mailbox.....	7-4
7.3.7	LISTMSG^XMXAIB(): Get a List of Messages in a Mailbox	7-6
7.3.8	NAMEBSKT^XMXAIB(): Change the Name of a Basket	7-9
7.3.9	QBSKT^XMXAIB(): Get information on a basket	7-10
7.3.10	QMBOX^XMXAIB(): Query a Mailbox for New Messages... ..	7-10
7.3.11	RSEQBSKT^XMXAIB(): Resequence Messages in a Basket.....	7-11
7.3.12	TERMMBOX^XMXAIB(): Remove All Traces of a User from MailMan Globals	7-11
8.0	Cross-category Activities—Mailboxes, Baskets, and Messages	8-1
8.1	^XM	8-1
8.1.1	\$\$NU^XM: Get Number of New Messages	8-1
8.1.2	^XMA	8-2
8.1.2.1	REC^XMA: Read/Manage Messages (Interactive)	8-2
8.1.3	^XMA0	8-2
8.1.3.1	ENTPRT^XMA0: Print a Message (Interactive)	8-2
8.1.3.2	HDR^XMA0: Headerless Print a Message	8-3
8.1.3.3	PR2^XMA0: Print a Message	8-4
8.1.4	^XMA1B.....	8-4
8.1.4.1	KL^XMA1B: Delete a Message from a Basket	8-4
8.1.4.2	KLQ^XMA1B: Delete a Message from a Basket (into "WASTE" basket) 8-5	
8.1.4.3	S2^XMA1B: Put a Message in a Basket.....	8-6
9.0	Mail Group Actions.....	9-1
9.1	^XMA21	9-1
9.1.1	CHK^XMA21: Verify User's Mail Group Membership	9-1
9.2	^XMBGRP	9-1
9.2.1	\$\$DM^XMBGRP(): Delete Local Members from a Mail Group	9-1
9.2.2	\$\$MG^XMBGRP(): Create New Mail Group or Add Members to an Existing Mail Group	9-2
9.3	^XMXAIG.....	9-3
9.3.1	ADDMBRS^XMXAIG(): Add Member(s) to Mail Group(s)	9-3
9.3.2	DROP^XMXAIG(): Drop Member from a Mail Group	9-3
9.3.3	\$\$GOTLOCAL^XMXAIG(): Check if a Mail Group has <i>Active</i> Local Members	9-4
9.3.4	JOIN^XMXAIG(): Enable User to Enroll in (Join) a Mail Group.....	9-5

10.0	Bulletins—Creating and Sending	10-1
10.1	^XMB	10-1
10.1.1	^XMB: Create & Send a Bulletin in the Background	10-1
10.1.2	BULL^XMB: Create & Send a Bulletin (Interactive)	10-2
10.1.3	EN^XMB: Create & Send a Bulletin in the Foreground.....	10-2
11.0	Address Lookup	11-1
11.1	^XMA21	11-1
11.1.1	DES^XMA21: Address Lookup (Interactive, Next Default Recipient List) 11-1	
11.1.2	DEST^XMA21: Address Lookup (Interactive, First Default Recipient List).....	11-2
11.1.3	INST^XMA21: Address Lookup (Non-Interactive)	11-2
11.1.4	WHO^XMA21: Address Lookup (Non-Interactive).....	11-3
12.0	User Information	12-1
12.1	^XMVVITAE.....	12-1
12.1.1	INIT^XMVVITAE(): Set Up Vital User Information	12-1
12.1.2	OTHER^XMVVITAE: Change User Settings When User Becomes a Surrogate.....	12-3
12.1.3	SELF^XMVVITAE: Restore Certain MailMan Settings Once User No Longer a Surrogate.....	12-4
13.0	User Actions—Interactive	13-1
13.1	^XM	13-1
13.1.1	CHECKIN^XM: Entry Action for Any Subordinate MailMan Option 13-1	
13.1.2	CHECKOUT^XM: Exit Action for Any Subordinate MailMan Option 13-1	
13.1.3	EN^XM: Entry Action of the Primary MailMan Option—Set Up Environment	13-1
13.1.4	HEADER^XM: Entry Action of the Primary MailMan Option—Display User Greeting	13-2
13.2	^XMXAPIU	13-2
13.2.1	READ^XMXAPIU: Read/Manage messages in a mailbox.....	13-2
13.2.2	READNEW^XMXAPIU: Read New Messages in a Mailbox ..	13-3
13.2.3	SEND^XMXAPIU: Send a Message	13-3
13.2.4	TOWHOM^XMXAPIU(): Ask User for Message Addressees.	13-4
14.0	Security—Permissions and Restrictions	14-5
14.1	Errors.....	14-5
14.2	^XMXSEC.....	14-5
14.2.1	\$\$ACCESS^XMXSEC(): Check if User Can Access a Message	14-5
14.2.2	\$\$ANSWER^XMXSEC(): Check if User Can Answer a Message	14-5
14.2.3	\$\$BCAST^XMXSEC(): Check if Message was Broadcast.....	14-6
14.2.4	\$\$CLOSED^XMXSEC(): Check if Message is "Closed"	14-6
14.2.5	\$\$CONFID^XMXSEC(): Check if Message is "Confidential" .	14-6

14.2.6	\$\$CONFIRM^XMXSEC(): Check if Message is "Confirm Receipt Requested"	14-7
14.2.7	\$\$COPY^XMXSEC(): Check if User Can Copy a Message ..	14-7
14.2.8	\$\$DELETE^XMXSEC(): Check if User Can Delete/Terminate a Message	14-7
14.2.9	\$\$FORWARD^XMXSEC(): Check if User Can Forward a Message 14-8	
14.2.10	\$\$INFO^XMXSEC(): Check if Message is "Information Only"	14-8
14.2.11	\$\$LATER^XMXSEC(): Check if User Can "Later" a Message	14-8
14.2.12	\$\$MOVE^XMXSEC(): Check if User Can Save or Filter a Message 14-9	
14.2.13	\$\$ORIGIN8R^XMXSEC(): Check if User Sent a Message....	14-9
14.2.14	\$\$POSTPRIV^XMXSEC: Check if User has Postmaster Privileges 14-9	
14.2.15	\$\$PRIORITY^XMXSEC(): Check if Message is "Priority"....	14-10
14.2.16	\$\$READ^XMXSEC(): Check if User Can Read a Message	14-10
14.2.17	\$\$REPLY^XMXSEC(): Check if User Can Reply to a Message	14-11
14.2.18	\$\$RPRIV^XMXSEC(): Check if Surrogate has READ Privileges	14-11
14.3	\$\$RWPRIV^XMXSEC(): Check if Surrogate has READ or WRITE Privileges	14-11
14.3.1	\$\$SEND^XMXSEC(): Check if User Can Send a Message.	14-12
14.3.2	\$\$SURRACC^XMXSEC(): Check if Surrogate Can Access a Message 14-12	
14.3.3	\$\$SURRCONF^XMXSEC(): Check if Message is "Confidential" & Surrogate Access	14-12
14.3.4	\$\$WPRIV^XMXSEC: Check if Surrogate has WRITE Privileges	14-13
14.3.5	\$\$ZCLOSED^XMXSEC(): Check if Message is "Closed"....	14-13
14.3.6	\$\$ZCONFID^XMXSEC(): Check if Message is "Confidential"	14-14
14.3.7	\$\$ZCONFIRM^XMXSEC(): Check if Message is "Confirm Receipt Requested"	14-14
14.3.8	\$\$ZINFO^XMXSEC(): Check if Message is "Information Only"	14-14
14.3.9	\$\$ZORIGIN8^XMXSEC(): Check if User Sent a Message ..	14-15
14.3.10	\$\$ZPOSTPRV^XMXSEC: Check if User has Postmaster Privileges 14-15	
14.3.11	\$\$ZPRI^XMXSEC(): Check if Message is "Priority".....	14-16
14.4	^XMXSEC1	14-16
14.4.1	CHKLINES^XMXSEC1(): Check if Message is Too Long to be Sent to a Remote Site.....	14-16
14.4.2	CHKMSG^XMXSEC1(): Check Message Location	14-17
14.4.3	\$\$COPYAMT^XMXSEC1(): Check Total Number of Lines & Responses to be Copied	14-17
14.4.4	\$\$COPYLIMS^XMXSEC1: Get Message Copy Limits	14-17
14.4.5	\$\$COPYRECP^XMXSEC1(): Check Total Number of Recipients on a Message	14-18
14.4.6	GETRESTR^XMXSEC1(): Get Sending/Forwarding Message Restrictions.....	14-18

14.4.7	OPTGRP^XMSEC1(): Determine User Capabilities at Basket/Message Group Level.....	14-19
14.4.8	\$PAKMAN^XMSEC1(): Check if PackMan Message.....	14-20
14.4.9	\$\$\$PRIV^XMSEC1: Check if User Authorized to Conduct Super Search.....	14-20
14.4.10	\$\$\$SRIV^XMSEC1: Check if User Authorized to Conduct Super Search.....	14-21
14.5	^XMSEC2.....	14-21
14.5.1	\$\$EDIT^XMSEC2(): Check if User Can Edit a Message...	14-21
14.5.2	OPTEDIT^XMSEC2(): Determine What the User Edit	14-21
14.5.3	OPTMSG^XMSEC2(): Determine What the User Can do with a Message.....	14-22
15.0	Servers—Message Activities.....	15-1
15.1.1	^XMA1C	15-1
15.1.1.1	REMSBMSG^XMA1C: Delete a Message from a Server Basket	15-1
15.1.1.2	SETSB^XMA1C: Put a Message in a Server Basket	15-1
15.1.2	^XMS1	15-2
15.1.2.1	\$\$\$SRVTIME^XMS1(): Set Server-related Fields in the Message File	15-2
15.1.2.2	\$\$\$STATUS^XMS1(): Get Status of a Server Recipient...	15-3
16.0	Utilities—General Development	16-1
16.1.1	^XM	16-1
16.1.1.1	^XM: Direct Entry Into MailMan (Without Menus).....	16-1
16.1.1.2	KILL^XM: MailMan Variable Cleanup.....	16-1
16.1.1.3	N1^XM: Create a Mailbox for a User.....	16-1
16.1.1.4	NEW^XM: Create a Mailbox for a User	16-2
16.1.2	^XMADGO.....	16-3
16.1.2.1	ZTSK^XMADGO: Start Tasks to Deliver Messages in Local Delivery Queues	16-3
16.1.3	^XMCTLK	16-3
16.1.3.1	GO^XMCTLK: Display Keyboard & Data Entries (Interactive)	16-3
16.1.4	^XMCU1	16-3
16.1.4.1	\$\$DECODEUP^XMCU1(): Convert ~U~ to ^ in a String .	16-3
16.1.4.2	\$\$ENCODEUP^XMCU1(): Convert ^ to ~U~ in a String .	16-4
16.1.4.3	\$\$RTRAN^XMCU1(): Undo \$\$STRAN^XMCU1 Conversion	16-4
16.1.4.4	\$\$STRAN^XMCU1(): Convert Control Characters to Printable Characters in a String.....	16-4
16.1.5	^XMUT7.....	16-5
16.1.5.1	ENT^XMUT7(): Send a Test Message to a User's Forwarding Address	16-5
17.0	Utilities—Messages and Mailboxes	17-1
17.1	^XMUTIL	17-1
17.1.1	\$\$BMSGCT^XMUTIL(): Get the Number of Messages in a User's Basket	17-1

17.1.2	\$\$BNMSGCT^XMXUTIL(): Get the Number of New Messages in a User's Basket	17-1
17.1.3	\$\$BPMMSGCT^XMXUTIL(): Get the Number of New Priority Messages in a User's Basket.....	17-1
17.1.4	\$\$BSKTNAME^XMXUTIL(): Get the Name of a User's Basket	17-1
17.1.5	KVAPOR^XMXUTIL(): Set/Remove a Message Vaporize Date in a User's Basket	17-2
17.1.6	LASTACC^XMXUTIL(): Record that the User has Read the Message	17-2
17.1.7	MAKENEW^XMXUTIL(): Make a Message New & Update the New Message Counts.....	17-3
17.1.8	\$\$NAME^XMXUTIL(): Get the User's Name, Title, and/or Institution	17-3
17.1.9	\$\$NETNAME^XMXUTIL(): Get User's Network Name & Domain	17-4
17.1.10	\$\$NEWS^XMXUTIL(): Get Information on New Messages in a User's Mailbox	17-4
17.1.11	NONEW^XMXUTIL(): Make a Message <i>Not</i> New & Update the New Message Counts.....	17-4
17.1.12	PAGE^XMXUTIL(): Display "Continue" Prompt to User	17-5
17.1.13	\$\$TMSGCT^XMXUTIL(): Get the Total Number of Messages in a User's Mailbox	17-5
17.1.14	\$\$TNMSGCT^XMXUTIL(): Get the Total Number of New Messages in a User's Mailbox	17-5
17.1.15	\$\$TPMSGCT^XMXUTIL(): Get the Total Number of New Priority Messages in a User's Mailbox.....	17-6
17.1.16	WAIT^XMXUTIL(): Display "Continue" Prompt to User	17-6
18.0	Utilities—Dates and Strings.....	18-1
18.1.1	^XMXUTIL1	18-1
18.1.1.1	\$\$CONVERT^XMXUTIL1(): Convert Internet Date/Time to VA FileMan Date/Time	18-1
18.1.1.2	\$\$CTRL^XMXUTIL1(): Strip Control Characters from a String	18-1
18.1.1.3	\$\$DECODEUP^XMXUTIL1(): Convert All ~U~ to ^ in a String	18-1
18.1.1.4	\$\$ENCODEUP^XMXUTIL1(): Convert All ^ to ~U~ in a String	18-2
18.1.1.5	\$\$GMTDIFF^XMXUTIL1(): Get the +-hhmm Difference from Greenwich Mean Time (GMT)	18-2
18.1.1.6	\$\$INDT^XMXUTIL1(): Convert VA FileMan Date/Time to Internet Date/Time	18-2
18.1.1.7	\$\$MAXBLANK^XMXUTIL1(): Reduce Consecutive Spaces in a String	18-2
18.1.1.8	\$\$MELD^XMXUTIL1(): Combine a String & Number to Form a New String of a Given Length.....	18-3
18.1.1.9	\$\$MMDT^XMXUTIL1(): Reformat VA FileMan Date	18-3

18.1.1.10	\$\$SCRUB^XMXUTIL1(): Strip Control Characters & Leading/Trailing Spaces from a String.....	18-4
18.1.1.11	\$\$STRIP^XMXUTIL1(): Strip Leading/Trailing Spaces from a String	18-4
18.1.1.12	\$\$TIMEDIFF^XMXUTIL1(): Reformat Decimal Time Difference to +-hhmm	18-4
18.1.1.13	\$\$TSTAMP^XMXUTIL1: Get a Timestamp (\$H Expressed in Seconds)	18-5
18.1.1.14	ZONEDIFF^XMXUTIL1(): Get Time Difference Between Time Zone and Local Time	18-5
19.0	Utilities—Message Information	19-6
19.1	^XMXUTIL2	19-6
19.1.1	\$\$BSKT^XMXUTIL2(): Get Basket Information	19-6
19.1.2	\$\$DATE^XMXUTIL2(): Get Message Sent Date	19-6
19.1.3	\$\$FROM^XMXUTIL2(): Get Message From Information	19-7
19.1.4	INMSG^XMXUTIL2(): Get Message Information	19-7
19.1.5	INMSG1^XMXUTIL2(): Get Message Information (Part 1)	19-9
19.1.6	INMSG2^XMXUTIL2(): Get Message Information (Part 2) ..	19-10
19.1.7	INRESP^XMXUTIL2: Get Response Information	19-11
19.1.8	INRESPS^XMXUTIL2(): Get Message Response Information	19-12
19.1.9	\$\$KSEQN^XMXUTIL2(): Get Message Sequence Number	19-12
19.1.10	\$\$LINE^XMXUTIL2(): Get Number of Text Lines in a Message	19-13
19.1.11	\$\$NEW^XMXUTIL2(): Get New Message Indicator	19-13
19.1.12	\$\$PRI^XMXUTIL2(): Get Priority Message Indicator	19-13
19.1.13	\$\$QRESP^XMXUTIL2(): Check if Message is a Response	19-13
19.1.14	\$\$RESP^XMXUTIL2(): Get the Number of Responses to a Message	19-14
19.1.15	\$\$SUBJ^XMXUTIL2(): Get Message Subject.....	19-14
19.1.16	\$\$ZDATE^XMXUTIL2(): Get Message Sent Date	19-14
19.1.17	\$\$ZFROM^XMXUTIL2(): Get Message From Information...	19-15
19.1.18	\$\$ZNODE^XMXUTIL2(): Get Message Zero Node	19-15
19.1.19	\$\$ZPRI^XMXUTIL2(): Get Priority Message Indicator.....	19-15
19.1.20	\$\$ZREAD^XMXUTIL2(): Get the Number of Responses Read	19-16
19.1.21	\$\$ZSUBJ^XMXUTIL2(): Get Message Subject	19-16
19.2	^XMXUTIL3	19-16
19.2.1	Q^XMXUTIL3(): List/Find Message Addressees	19-16
19.2.2	QD^XMXUTIL3(): List/Find Message Recipients.....	19-17
19.2.3	QL^XMXUTIL3(): List/Find "Latered" Message Addressees	19-20
19.2.4	QN^XMXUTIL3(): Get Network Message Header Records .	19-21
19.2.5	QX^XMXUTIL3(): Local Recipient Extract	19-22
20.0	Glossary	1
21.0	Appendix A—Message Server Protocol	1
21.1	Appendix B—Efficient Use of the API.....	1
22.0	Appendix C—Looking Up Messages	1

23.0	Appendix D—Setting Up Bulletins	1
23.1	Index.....	1

Figures and Tables

Figure D-1. An Example of a Bulletin.....	1
Figure D-2. Sample code to call to the Bulletin API.....	1
Figure D-3. An Example of Setting Up a Bulletin Cross-reference (1 of 2).....	3
Figure D-4. An Example of Setting Up a Bulletin Cross-reference (2 of 2).....	4

Orientation

This *MailMan Developer's Guide* is intended for use in conjunction with Veterans Health Information Systems and Technology Architecture (VistA) MailMan. It outlines programmer details of the VistA MailMan software (e.g., Application Program Interfaces [APIs] and Direct Mode Utilities) and gives guidelines on how the software is used within VistA.

The intended audience of this manual is all primary (key) stakeholders. The primary stakeholders include:

- VistA Infrastructure and Security Services (ISS) Development Team.
- Other VistA project development teams and programmers.
- Information Resource Management (IRM) personnel responsible for maintaining MailMan.
- Enterprise VistA Support (EVS).

How to Use this Manual



Throughout this manual, advice and instructions are offered regarding the use of MailMan V. 8.0 and the functionality it provides for Veterans Health Information Systems and Technology Architecture (VistA) software products. This manual discusses the use of MailMan's Application Program Interfaces (APIs) AND Direct Mode Utilities.

There are no special legal requirements involved in the use of MailMan.

This manual uses several methods to highlight different aspects of the material:

- Various symbols are used throughout the documentation to alert the reader to special information. The following table gives a description of each of these symbols:

Table ii. Documentation symbol descriptions

Symbol	Description
	NOTE/REF: Used to inform the reader of general information including references to additional reading material.
	CAUTION or DISCLAIMER: Used to inform the reader to take special notice of critical information.

- Descriptive text is presented in a proportional font (as represented by this font).
- Conventions for displaying TEST data in this document are as follows:

- The first three digits (prefix) of any Social Security Numbers (SSN) will begin with either "000" or "666".
- Patient and user names will be formatted as follows: [Application Name]PATIENT,[N] and [Application Name]USER,[N] respectively, where "Application Name" is defined in the Approved Application Abbreviations document and "N" represents the first name as a number spelled out and incremented with each new entry. For example, in Kernel (KRN) test patient and user names would be documented as follows: KRNPATIENT,ONE; KRNPATIENT,TWO; KRNPATIENT,THREE; etc.
- Sample HL7 messages, "snapshots" of computer online displays (i.e., roll-and-scroll screen or character-based screen captures/dialogues) and computer source code, if any, are shown in a *non*-proportional font and enclosed within a box.
 - User's responses to online prompts will be boldface.
 - References to "<Enter>" within these snapshots indicate that the user should press the **Enter** key on the keyboard. Other special keys are represented within < > angle brackets. For example, pressing the **PF1** key can be represented as pressing <PF1>.
 - Author's comments, if any, are displayed in italics or as "callout" boxes.



NOTE: Callout boxes refer to labels or descriptions usually enclosed within a box, which point to specific areas of a displayed image.



NOTE: Unless otherwise noted, all sample screen captures/dialogue boxes in this manual are derived from using either MailMan's Detailed or Summary Full Screen message readers.

- This manual refers in many places to the M programming language. Under the 1995 American National Standards Institute (ANSI) standard, M is the primary name of the M programming language, and M will be considered an alternate name. This manual uses the name M.
- Descriptions of direct mode utilities are prefaced with the standard M ">" prompt to emphasize that the call is to be used *only in direct mode*. They also include the M command used to invoke the utility. The following is an example:

```
>D ^XUP
```

- The following conventions will be used with regards to APIs:
 - Headings for programmer API descriptions (e.g., supported for use in applications and on the Database Integration Committee [DBIC] list) include the routine tag (if any), the caret ("^") used when calling the routine, and the routine name. The following is an example:

```
TAG^ROUTINE
```


- For APIs that take input parameter, the input parameter will be labeled "required" when it is a required input parameter and labeled "optional" when it is an optional input parameter.
- For APIs that take parameters, parameters are listed in lowercase. This is to convey that the listed parameter name is merely a placeholder; M allows you to pass a variable of any name as the parameter or even a string literal (if the parameter is not being passed by reference). The following is an example of the formatting for input parameters:
TAG^ROUTINE(param1,[.]param2[,param3])
- Rectangular brackets [] around a parameter are used to indicate that passing the parameter is optional. Rectangular brackets around a leading period [.] in front of a parameter indicate that you can optionally pass that parameter by reference.
- All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field and file names, and security keys (e.g., the XUPROGMODE key).

How to Obtain Technical Information Online

Exported file, routine, and global documentation can be generated through the use of Kernel, MailMan, and VA FileMan utilities.



NOTE: Methods of obtaining specific technical information online will be indicated where applicable under the appropriate topic.

Help at Prompts

VistA M Server-based software provides online help and commonly used system default prompts. Users are encouraged to enter question marks at any response prompt. At the end of the help display, you are immediately returned to the point from which you started. This is an easy way to learn about any aspect of the software.

In addition to the "question mark" help, you can use the Help (User/Group Info., etc.) menu option on the main MailMan Menu to access the MailMan Help Frames through the following options:

- New Features in MailMan
- General MailMan Information
- Questions and Answers on MailMan
- Manual for MailMan Users



REF: For more information on obtaining MailMan online help, please refer to Chapter 12, "Online Help Information" in the *MailMan User Guide*.

Obtaining Data Dictionary Listings

Technical information about VistA M Server-based files and the fields in files is stored in data dictionaries (DD). You can use the List File Attributes option on the Data Dictionary Utilities submenu in VA FileMan to print formatted data dictionaries.



REF: For details about obtaining data dictionaries and about the formats available, please refer to the "List File Attributes" chapter in the "File Management" topic of the *VA FileMan Advanced User Guide*.

Assumptions About the Reader

This manual is written with the assumption that the reader is familiar with the following:

- VistA computing environment:
 - Kernel—VistA M Server software
 - VA FileMan data structures and terminology—VistA M Server software
- Microsoft Windows environment
- M programming language

This manual provides an overall explanation of MailMan and the changes contained in MailMan V. 8.0. However, no attempt is made to explain how the overall VistA programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA home pages on the World Wide Web (WWW) and VA Intranet for a general orientation to VistA. For example, go to the Veterans Health Administration (VHA) Office of Information (OI) Health Systems Design & Development (HSD&D) Home Page at the following Intranet Web address:

<http://vista.med.va.gov/>

Reference Materials

Readers who wish to learn more about MailMan should consult the following:

- *MailMan Release Notes*
- *MailMan Installation Guide*
- *MailMan Getting Started Guide*
- *MailMan Developer's Guide (this manual)*
- *MailMan User Guide*
- *MailMan Network Reference Guide*
- *MailMan Package Security Guide*
- *MailMan Systems Management Guide*

- *MailMan Technical Manual*
- MailMan Home Page at the following Web address:
<http://vista.med.va.gov/mailman/index.asp>

This site contains other information and provides links to additional documentation.

VistA documentation is made available online in Microsoft Word format and in Adobe Acrobat Portable Document Format (PDF). The PDF documents *must* be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following Web address:

<http://www.adobe.com/>



REF: For more information on the use of the Adobe Acrobat Reader, please refer to the "Adobe Acrobat Quick Guide" at the following Web address:

<http://vista.med.va.gov/iss/acrobat/index.asp>

VistA documentation can be downloaded from the Health Systems Design and Development (HSD&D) VistA Documentation Library (VDL) Web site:

<http://www.va.gov/vdl/>

VistA documentation and software can also be downloaded from the Enterprise VistA Support (EVS) anonymous directories:

- Albany OIFO <ftp.fo-albany.med.va.gov>
- Hines OIFO <ftp.fo-hines.med.va.gov>
- Salt Lake City OIFO <ftp.fo-slc.med.va.gov>
- Preferred Method <download.vista.med.va.gov>

This method transmits the files from the first available FTP server.



DISCLAIMER: The appearance of external hyperlink references in this manual does not constitute endorsement by the Department of Veterans Affairs (VA) of this Web site or the information, products, or services contained therein. The VA does not exercise any editorial control over the information you may find at these locations. Such links are provided and are consistent with the stated purpose of this VA Intranet Service.

1.0 Introduction

MailMan has many documented Application Program Interfaces (APIs). In addition to the "classic" APIs, additional APIs have been created to support other front ends to MailMan (e.g., a Graphical User Interface [GUI]). Where possible, existing MailMan code has been altered to use the latest APIs.

The following topics are discussed in this manual:

- Creating/Sending/Forwarding Messages
- Editing Messages
- Message Actions
- Getting Information About and Text From Messages
- Replies/Answers to Messages—Creating and Sending
- Basket Actions
- Cross-category Activities—Mailboxes, Baskets, and Messages
- Mail Group Actions
- Bulletins—Creating and Sending
- Address Lookup
- User Information
- User Actions—Interactive
- Security—Permissions and Restrictions
- Servers—Message Activities
- Utilities—General Development
- Utilities—Messages and Mailboxes
- Utilities—Dates and Strings
- Utilities—Message Information



CAUTION: Application Program Interfaces *not* documented are subject to change and are *not* supported. Use them at your own risk!

Those VistA applications *not* using approved MailMan APIs or having an approved Database Integration Agreement (IA) with MailMan, *must* review their code to determine if modifications are necessary *prior* to installing any MailMan patches.

1.1 Common Variables

Often, MailMan APIs require the existence of certain variables when they are invoked. Some variables are understood to exist as they are set up during system security at signon through Kernel. These variables include:

- DUZ
- DTIME
- DT
- U
- IO
- IOST
- IOSL
- IOF

Normally, these variables should *not* be reset or KILLed.

The following table lists and briefly describes some of the common variables used in the MailMan APIs:

Table 1-1. Common variables

Variable	Description
DUZ	User's DUZ. If DUZ is not defined, it defaults to the Postmaster. This is who is really sending the message.
XMDUZ	User's DUZ or FREE TEXT. This is from whom the message will appear to be. If it is not defined, it defaults to DUZ. (If DUZ is not defined, it defaults to Postmaster.) If it is FREE TEXT, it must not be more than 70 characters.
XMSUB	Subject of the message. It should be from 3 to 65 characters in length. If it is less than 3 characters, then three dots ("...") will be appended to it. If it is more than 65 characters, then it will be truncated.
XMTEXT	The name of the array (in open format) containing the text of the message. The array itself can be a local or a global variable, and it must be in a format acceptable to VA FileMan word-processing fields.

Variable	Description
XMY	<p>Addressee array, XMY(x)="", where x can be: User's DUZ or enough of the user's name for a positive ID. For example: XMY(1301)=" OR XMY("lastname,first")=""</p> <p>If the user or SHARED,MAIL is an addressee, the basket to place the message in can be specified. For example: XMY(DUZ,0)=basket name or IEN</p> <p>If SHARED,MAIL is an addressee, the automatic delete date of the message can be specified. For example: XMY(.6,"D")=delete date in any format that VA FileMan understands.</p> <p>G.group name (enough for positive ID). For example: XMY("G.group name")=""</p> <p>S.server name (enough for positive ID).</p> <p>D.device name (enough for positive ID).</p> <p>Prefix the above (except devices and servers) by: I: for "Information Only" recipient (<i>cannot</i> reply). For example: XMY("I:1301")="" or XMY("I:lastname,first")=""</p> <p>C: for "Copy" recipient (not expected to reply). For example: XMY("C:1301")="" or XMY("C:lastname,first")=""</p> <p>L@datetime: for when (in future) to send to this recipient (datetime can be anything accepted by VA FileMan). For example: XMY("L@25 DEC@0500:1301")="" or XMY("L@1 JAN:lastname,first")="" or XMY("L@2981225.05:1301")=""</p> <p>(Can combine IL@datetime: or CL@datetime:)</p> <p>To delete any recipient (including users, groups, devices, and servers, prefix with a hyphen/dash ("-"). For example: XMY(-1301)=" or XMY("-lastname,first")=""</p> <p>To address any recipient (including users, groups, devices, and servers) at a remote site, just add the @site name. For example: XMY(recipient@site name)=""</p>
XMZ	Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

1.2 Errors

If any errors occur, the following variables may be defined:

XMERR The number of errors.

^TMP("XMERR", \$J, <error number>, "TEXT", <line number>)=<error text>

2.0 Creating/Sending/Forwarding Messages

2.1 ^XMA11

2.2 \$\$INFO^XMA11(): Edit "Information Only" Field

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	
Description	This extrinsic function is interactive. It lets the user edit message XMZ's "Information Only" field and returns 0.
Format	\$\$INFO^XMA11(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: 0

2.3 ^XMA11A

2.3.1 WRITE^XMA11A: Send a Message (Interactive)

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	1233
Description	This API sends a message interactively. It is the same as XMSEND, the Send a Message option. It is similar to ENT^XMD, but it does <i>not</i> display the MailMan greeting.
Format	WRITE^XMA11A

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables.
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. XMDUZ: (optional) User's DUZ.
Output Variables	None

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables.
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMDUZ: (optional) User's DUZ.

Output Variables None

2.4 ^XMA2

2.4.1 GET^XMA2: Create Message Stub

Reference Type Supported

Category Creating/Sending/Forwarding Messages (Classic MailMan)

IA # 10066

Description This API creates a new message stub in the MESSAGE file (#3.9). Unlike XMZ^XMA2, however, if the stub creation fails, your process will halt.



NOTE: Recommend you use XMZ^XMA2: Create Message Stub or CRE8XMZ^XMXAPI(xmsubj,.xmz).

Format GET^XMA2

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables.
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. If it is zero or null or not defined, it defaults to DUZ.

XMSUB: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

Output Variables XMZ: Message number in the MESSAGE file (#3.9), if stub creation succeeds.



CAUTION: This API works exactly the same way as XMZ^XMA2, except that if the stub creation fails, your process will HALT! Thus, this API should *not* be used.

Instead, use XMZ^XMA2: Create Message Stub or CRE8XMZ^XMXAPI(xmsubj,.xmz).

2.4.2 XMZ^XMA2: Create Message Stub

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10066
Description	This API creates a new message stub in the MESSAGE file (#3.9). A message stub is an entry in the MESSAGE file (#3.9) with no text or recipients. The SUBJECT field (#.01) (#.01):MESSAGE File (#3.9) will be set to XMSUB. The FROM field (#1) (#1):MESSAGE File (#3.9) is set to XMDUZ. The SENT DATE/TIME field (#1.4) is set to the current date/time, in internal VA FileMan format. If XMDUZ is a number and it differs from DUZ, then the SENDER field (#1.1) will be set to DUZ. If XMDUZ=.5 and DUZ=.5, the INFORMATION ONLY? field (#1.97) will be set to "y", thus, making the stub "Information only."
Format	XMZ^XMA2

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables.
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. If it is zero, null, or not defined, it defaults to DUZ</p> <p>XMSUB: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p>
Output Variables	<p>XMZ:Results: Message number in the MESSAGE file (#3.9)—If stub creation succeeds. -1—If stub creation fails. For example, if a lock on the MESSAGE file (#3.9) <i>cannot</i> be achieved.</p>



REF: Compare to the CRE8XMZ^XMXAPI(xmsubj,.x mz) API described in the "Message Actions—Building Block APIs" topic in Chapter 4, "Message Actions," in this manual.

Example

```
S XMSUB="TEST RESULTS",XMDUZ="TESTING SOFTWARE" D XMZ^XMA2
```

This creates a message stub from the TESTING SOFTWARE.

Once you have created a message stub and any time before you send the message, you can set other message type fields, by using VA FileMan as follows:

```
S DIE=3.9,DA=XMZ,DR="<field #>////<value>" D ^DIE
```

Table 2-1. Sample MESSAGE file (#3.9) field values

Field #	Value	Causes Message To Be
1.7	P	Priority
1.95	y	Closed
1.96	y	Confidential
1.97	y	Information only

For example, to force message 100213 to be priority and closed:

```
S DIE=3.9,DA=100213,DR="1.7////P;1.95////y" D ^DIE
```



REF: To find out how to add text to your message stub, please refer to the ENL^XMD: Add Text to a Message API in this chapter.

To find out how to forward your message to various recipients, please refer to the ENT1^XMD API in this chapter.

To find out how to add text to your message stub and send your message to various recipients, please refer to the ENT1^XMD: Forward a Message (Address Restrictions Waived) API in this chapter.

2.4.3 ^XMD: Create and Send a Message

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10070
Description	This API creates and sends a message. If there are no recipients, XMMG is set to "Error = No recipients." If no recipients are defined, and '\$D(ZTQUEUED)', then prompt for them. Addressing restrictions are waived. (It's as if you set XMDF.)



REF: Compare this API to the SENDMSG^XMXAPI(): Send a Message API described in Chapter 4, "Message Actions," in this manual


Format ^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMSUB: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMTEXT: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMY: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. If $\\$D(XMY) < 10$ (no recipients), and $\\$D(ZTQUEUED)$ (job running in the foreground), the user will be prompted for recipients. If there are no recipients, the message will be created, but it will not be sent, and XMMG will not be defined.</p>
Additional Input Variables	<p>XMMG: (optional) If there are no recipients in XMY and the job is running in the foreground, XMMG may contain the default recipient presented to the user. If XMMG is not defined, then the default recipient is the user.</p> <p>XMSTRIP: (optional) String containing characters that should be removed from the message text. The default is none.</p> <p>XMROU: (optional) Array of routines to be loaded in a PackMan message. For each routine, set $XMROU(x) = ""$, where x is the routine name.</p> <p>DIFROM: (optional) Specifically for the VA FileMan software.</p> <p>XMYBLOB: (optional) Specifically for the Imaging software.</p>
Output Variables	<p>XMZ: Results: Successful—Message number in the MESSAGE file (#3.9). Unsuccessful—Unchanged or undefined.</p> <p>XMMG: This is the variable that the calling program should check to determine whether or not the call was successful. Successful—Undefined. Unsuccessful—String containing error message.</p>
Variables KILLED Upon Exit	<p>If the call is successful, the following variables are KILLED: XMSUB, XMTEXT, XMY, XMSTRIP, XMMG, and XMYBLOB. If the call fails, those variables may or may not be KILLED, except for XMMG, which will contain an error string.</p>

 **NOTE:** When invoking ^XMD in pre-/post-init routine of the Kernel Installation and Distribution System (KIDS) build, the calling routine *must* NEW the DIFROM variable; otherwise, your message will not be delivered. As a rule, this process of NEW'ing is not specific to pre-/post-init routines in KIDS but to *all* routines that invoke ^XMD.

2.4.4 EN1^XMD: Add Text to a Message

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10070
Description	This API adds text to a message, addresses it, and sends it. If no recipients are defined, and $\$D(ZTQUEUED)$, then prompt for them.
Format	EN1^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..

- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables

DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMTEXT: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMDF: (optional) If \$D(XMDF) all addressing restrictions are waived.
XMY: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. If \$D(XMY)<10 (no recipients), and \$D(ZTQUEUED) (job running in the foreground), the user will be prompted for recipients. If there are no recipients, the message will be created, but it will not be sent, and XMMG will not be defined.
XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Additional Input Variables

XMMG: (optional) If there are no recipients in XMY and the job is running in the foreground, XMMG may contain the default recipient presented to the user. If XMMG is not defined, then the default recipient is the user.
XMSTRIP: (optional) String containing characters that should be removed from the message text. The default is none.
DIFROM: (optional) Specifically for the VA FileMan software.



REF: To create and send a PackMan message with globals in it, please refer to the ^XMPPG API in this chapter.

Output Variables
Variables KILLED
Upon Exit

None
 XMTEXT, XMY, XMSTRIP, XMMG



REF: Compare to the CRE8XMZ^XMXAPI(xmsubj,xmz) API described in the "Message Actions—Building Block APIs" topic in Chapter 4, "Message Actions," in this manual.

2.4.5 ENL^XMD: Add Text to a Message

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10070
Description	This API adds text to a message.
Format	ENL^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. XMTEXT: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Additional Input Variables	XMSTRIP: (optional) String containing characters that should be removed from the message text. The default is none.
Output Variables	None
Variables KILLed Upon Exit	XMSTRIP

2.4.6 ENT^XMD: Send a Message (Interactive)

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10070
Description	This API sends a message interactively. It is the same as the Send a Message option [XMSEND]. This call can be placed in a menu option as follows: Entry action: S XMMENU(0)=<name of the menu option> Routine: ENT^XMD Exit action: K XMMENU D CHECKOUT^XM



REF: Compare this API to the SENDMSG^XMXAPI(): Send a Message API described in Chapter 4, "Message Actions," in this manual.

Format ENT^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
Output Variables	None

2.4.7 ENT1^XMD: Forward a Message (Address Restrictions Waived)

Reference Type Supported

Category Creating/Sending/Forwarding Messages (Classic MailMan)
IA # 10070
Description This API forwards a message. Addressing restrictions are waived. (It's as if you set XMDF.)



REF: Compare this API to the FWDMSG^XMXAPI(): Forward Messages from a Basket API described in Chapter 4, "Message Actions," in this manual.

Format ENT1^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task

Input Variables DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMY: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Variables None
Variables KILLed Upon Exit

2.4.8 ENT2^XMD: Forward a Message

Reference Type Supported
Category Creating/Sending/Forwarding Messages (Classic MailMan)
IA # 10070
Description This API forwards a message. If '\$D(ZTQUEUEUED)', prompt for additional recipients, whether or not any are already defined.
Format ENT2^XMD

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ:	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	XMDUZ:	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	X MDF:	(optional) If \$D(X MDF) all addressing restrictions are waived.
	XMY	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
		If '\$D(ZTQUEUEUED) (job running in the foreground), the user will be prompted for additional recipients.
	XMZ	(required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None	
Variables KILLED Upon Exit	XMDUZ, XMY	

2.5 ^XMGAPI0

2.5.1 \$\$SUBCHK^XMGAPI0(): Validate Message Subject

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	1142
Description	This extrinsic function validates a message subject and returns either the valid subject or error string explaining why it's not valid. Leading and trailing spaces are automatically removed. The carets ("^") are automatically converted to ~U~.
Format	\$\$SUBCHK^XMGAPI0(xmsub,xmflg)
Input Parameters	xmsub: (required) Message subject. xmflg: (required) Interactive? 0—No 1—Yes

Output returns: Possible results, in actual order:

Subject is too long
 Non-interactive: 3-Entered subject too long...[^]\$E(<subject>,1,65)
 Interactive: "Entered subject too long..."
 1[^]\$E(<subject>,1,250)

At this point, leading and trailing spaces are removed, and carets ("[^]") are converted to ~U~.

Subject contains control characters
 Non-interactive: 5-Subject *cannot* contain control characters.[^]<subject>
 Interactive: "Control characters removed (<subject> is Subject accepted)." (Control characters are removed and checking continues.)

Subject is null
 Non-interactive: <subject>
 Interactive: <subject>

Subject is "?"
 Non-interactive: 4-Enter a Message Subject, between 3 & 65 characters long or '^' to exit.[^]<subject>
 Interactive: "Enter a Message Subject, between 3 & 65 characters long or '^' to exit."
 1[^]<subject>

Subject is too short
 Non-interactive: 1-SUBJECT must be at least 3 characters long.[^]<subject>
 Interactive: "SUBJECT must be at least 3 characters long."
 1[^]<subject>

Subject is reserved format
 Non-interactive: 2-Subject names of this format (1""R""1.N) are RESERVED [^]<subject>
 Interactive: "Subject names of this format (1""R""1.N) are RESERVED"
 1[^]<subject>

Subject is OK
 Non-interactive: [^]<subject>
 Interactive: [^]<subject>



REF: Compare this API to the `VSUBJ^XMXPAPI (.xmsubj)` API described in Chapter 4, "Message Actions," in this manual.

2.6 ^XMPG

2.6.1 ENT[^]XMPG: Create/Send PackMan Message with Globals

Reference Type	Supported
Category	Creating/Sending/Forwarding Messages (Classic MailMan)
IA #	10071

Description	This API creates and sends a PackMan message with globals in it. If no recipients are defined, the message will be created, but it will not be sent anywhere. Addressing restrictions are waived. (It's as if you set XMDF.) This API checks to ensure that the user who is using it has an Access code and a mailbox.
Format	ENT^XMPG

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables	DUZ:	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	XMDUZ:	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	XMSUB:	(required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	XMTEXT:	(required) String of open global roots, separated by semicolons. The globals are loaded into the PackMan message.
	XMY:	(optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
	^TMP("XMP", \$J):	(optional) Text to be placed in the PackMan message <i>must</i> be in the following format: ^TMP("XMP", \$J, i, 0) = <text>
Input Variables	DIFROM:	(optional) Specifically for the VA FileMan or KIDS software.
Output Variables	XMZ:	Results: Successful—Message number in the MESSAGE file (#3.9). Unsuccessful—Unchanged or undefined.
	XMMG:	Results: Successful—Unchanged or undefined. Unsuccessful—String containing error message.
Variables KILLED Upon Exit	XMY, ^TMP("XMP", \$J)	



NOTE: To create and send a PackMan message with routines in it, use the XMZ^XMA2: Create Message Stub API described in this chapter.

3.0 Editing Messages

These entry points edit different parts of a message. They do *not* perform any checks to see whether it is appropriate to do so. That is the responsibility of the calling routine.

Generally, these entry points expect that the input parameters are correct. They also expect that the calling application has assumed a level of responsibility and already taken care of the following:

- `INIT^XMOVITAE`: Has been called to set up the user's XMV arrays:, with vital user information, user preferences, and, if the user is a surrogate, determining level of authorization.
- Determined that the user is authorized to see the message. If the message is in the user's mailbox, then that's enough. Otherwise, `$$ACCESS^XMXSEC` should be used to determine authorization.
- `OPTMSG^XMXSEC2`: Has been called and has given its permission to edit the message or to toggle Information Only.



REF: The `$$EDIT^XMXSEC2` API also lets you know whether the user can edit the message.

- `OPTEDIT^XMXSEC2`: Has been called and has given its permission to edit the particular thing we are editing here.
- `INMSG2^XMXUTIL2`: Has been called to set `XMINSTR`. These routines expect that `XMINSTR` has been correctly set. They will change `XMINSTR` according to the item being edited.

3.1 ^XMXEDIT

3.2 CLOSED^XMXEDIT(): "Close" Flag Toggle

Reference Type	Supported
Category	Editing Messages
IA #	2730
Description	This API toggles the message's "Closed" flag. It sets <code>XMERR</code> and <code>^TMP("XMERR", \$J)</code> , if an error occurs.
Format	<code>CLOSED^XMXEDIT(xmz,.xminstr,.xmmsg)</code>
Input Parameters	<code>xmz::</code> (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). <code>.xminstr</code> : (required) Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"

Output Parameters .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"
 .xmmsg: Appropriate message, suitable for display to the user.

3.2.1 CONFID^MXEDIT(): "Confidential" Flag Toggle

Reference Type Supported
Category Editing Messages
IA # 2730
Description This API toggles the message's "Confidential" flag. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.
Format CONFID^MXEDIT(xmz, xminstr, xmmsg)
Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 .xminstr: (required) Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"
Output Parameters .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"
 .xmmsg: Appropriate message, suitable for display to the user.

3.2.2 CONFIRM^MXEDIT(): "Confirm Receipt Requested" Flag Toggle

Reference Type Supported
Category Editing Messages
IA # 2730
Description This API toggles the message's "Confirm Receipt Requested" flag. It does *not* set XMERR and ^TMP("XMERR", \$J).
Format CONFIRM^MXEDIT(xmz, xminstr, xmmsg)
Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 .xminstr: (required) Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"
Output Parameters .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"
 .xmmsg: Appropriate message, suitable for display to the user.

3.2.3 DELIVER^MXEDIT(): Set/Delete Message Delivery Basket (All Users)

Reference Type Supported
Category Editing Messages
IA # 2730
Description This API sets/deletes the message delivery basket for all users. It does *not* set XMERR and ^TMP("XMERR", \$J).
Format DELIVER^MXEDIT(xmz, xmdbst, xminstr, xmmsg)

Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>xmdbskt: (required) New Delivery basket name: "@" (at-sign), if you want to delete it.</p>
Output Parameters	<p>.xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: ("RCPT BSKT") Set to XMDBSKT or KILLED if XMDBSKT="@".</p> <p>.xmmsg: Appropriate message, suitable for display to the user.</p>

3.2.4 INFO^XMXEDIT(): "Information Only" Flag Toggle

Reference Type	Supported
Category	Editing Messages
IA #	2730
Description	This API toggles the message's "Information Only" flag. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).
Format	INFO^XMXEDIT(xmz,.xminstr,.xmmsg)
Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>.xminstr: (required) Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"</p>
Output Parameters	<p>.xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"</p> <p>.xmmsg: Appropriate message, suitable for display to the user.</p>

3.2.5 PRIORITY^XMXEDIT(): "Priority" Flag Toggle

Reference Type	Supported
Category	Editing Messages
IA #	2730
Description	This API toggles the message's "Priority" flag. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).
Format	PRIORITY^XMXEDIT(xmz,.xminstr,.xmmsg)
Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>.xminstr: (required) Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"</p>
Output Parameters	<p>.xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: "FLAGS"</p> <p>.xmmsg: Appropriate message, suitable for display to the user.</p>

3.2.6 SUBJ^XMXEDIT(): Change Message Subject

Reference Type	Supported
Category	Editing Messages
IA #	2730

Description This API changes the message subject. It does not set XMERR and ^TMP("XMERR", \$J).

Format SUBJ^XMXEDIT(xmz,xmsubj,.xmim)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
xmsubj: (required) Subject of the message. It must be from 3 to 65 characters in length. If null, it defaults to "* No Subject *". If the subject is "* No Subject *" and the message is sent to a remote site, the subject in the "SUBJECT:" header record will be null.

Output Parameters .xmim: Message information: ("SUBJ") Message subject.

3.2.7 TEXT^XMXEDIT(): Replace Message Text

Reference Type Supported

Category Editing Messages

IA # 2730

Description This API replaces the message text. It does not set XMERR and ^TMP("XMERR", \$J).

Format TEXT^XMXEDIT(xmz,xmbody)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
xmbody: (required) Closed root of array containing new message text. The root cannot be called "XMBODY". Also, it must be VA FileMan word-processing compatible.

Output None

3.2.8 VAPOR^XMXEDIT(): Set/Delete Message Vaporize Date

Reference Type Supported

Category Editing Messages

IA # 2730

Description This API sets/deletes the message vaporize date. It does *not* set XMERR and ^TMP("XMERR", \$J).



REF: This routine does not set the message vaporize date in a user's basket. Use the KVAPOR^XMXUTIL API to do that.

Format VAPOR^XMXEDIT(xmz,xmvapor,.xminstr,.xmmsg)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
xmvapor: (required) New message vaporize date/time. The date *must* be in VA FileMan format. "@" (at-sign), if you want to delete it.

Output Parameters .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" list in Chapter 4, "Message Actions," in this manual: ("VAPOR")
Set to XMVAPOR or KILLED if XMVAPOR="@".
.xmmsg Appropriate message, suitable for display to the user.

4.0 Message Actions

4.1 Parameter Definitions

Table 4-1. Parameter definitions—Message actions

Parameter	Description	Meaning
xminstr	(optional) Array of special instructions:	
	("ADDR FLAGS")	Special addressing instructions, any or all of the following: I—Do not Initialize (KILL) the ^TMP addressee global, because it already contains addressees for this message, as a result of a previous call to an API. R—Do not Restrict message addressing: - Ignore "domain closed." - Ignore "keys required for domain." - Ignore "may not forward to domain." - Ignore "may not forward priority mail to groups." - Ignore "may not forward priority mail to groups." - Ignore "message length restrictions to remote addressees." X—Do not create the ^TMP addressee global, because addressees are only being checked for validity.
	("FDATE")	Add users to messages originating on or after this date. Must be any exact date recognized by VA FileMan. The default is from the beginning of time. It is used in conjunction with FLAGS.
	("FLAGS")	Message is any or all of the following: P—Priority. I—Information Only (<i>cannot</i> be replied to). X—Closed message (<i>cannot</i> be forwarded). C—Confidential message (surrogate <i>cannot</i> read). S—Send to sender (make sender a recipient). R—Confirm receipt (return receipt requested). F—Forward messages to users, if the users aren't already on the messages.

Parameter	Description	Meaning
	("FROM")	String saying who the message is from. The default is user. This string is placed in the FROM field (#1) in the MESSAGE file (#3.9). It must <i>not</i> be any real person, except for the Postmaster. The DUZ is <i>not</i> captured in the SENDER field (#1.1) in the MESSAGE file (#3.9); thus, making this option well suited for messages from VistA software.
	("FWD BY")	String saying who forwarded the message. The default is user. This string is placed in the FORWARDED BY field (#8) in the RECIPIENT Multiple field in the MESSAGE file (#3.9). It must <i>not</i> be any real person, except for the Postmaster. The DUZ is <i>not</i> captured in FORWARDED BY field (#8) , "FORWARDED BY (XMDUZ)" in the RECIPIENT Multiple field of the MESSAGE file (#3.9); thus, making this option well-suited for messages forwarded by VistA software.
	("HDR")	Print the messages with a header? - 1 (default)—Yes - 0—No
	("LATER")	Date/time (any format understood by VA FileMan) on which to send this message. The default is now.
	("NET REPLY")	Should reply be sent over the network? - 1 (default)—Yes - 0—No Currently, only valid if sender of original message is remote.
	("NET SUBJ")	Subject of reply to be sent over the network. The default is: "Re: <subject of original message>" Ignored unless XMINSTR("NET REPLY")=1.
	("RCPT BSKT")	Basket to deliver to for all recipients. The default is the "IN" basket. Recipients must have specified in their personal preferences that such targeted basket delivery is allowed. Otherwise, this option is ignored.
	("RECIPS")	Print recipients along with the message? - 0 (default) - 1— Print summary recipients - 2—Print detailed recipients.

Parameter	Description	Meaning
	("RESPS")	Print which responses? * (default)—Original message and all responses. 0—Original message only. Range list (e.g., "0-3,5,7-99")—Ignored if more than one message is printed. This parameter is not checked. It <i>must</i> be correct. Range list can also be open-ended (e.g., "1,2,5-" means print responses 1, 2, and responses 5 to the end)..
	("SCR KEY")	Scramble key (implies that message should be scrambled). It <i>must</i> be from 3 to 20 characters in length.
	("SCR HINT")	Hint for scramble key (mandatory if message is to be scrambled). It <i>must</i> be from 1 to 40 characters in length.
	("SELF BSKT")	Basket to deliver to, if sender is recipient. The default is the "IN" basket.
	("SHARE BSKT")	Basket to deliver to if SHARED,MAIL is recipient. The default is the "IN" basket.
	("SHARE DATE")	Date/time (any format understood by VA FileMan) to delete this message from SHARED,MAIL if SHARED,MAIL is the recipient.
	("STRIP")	String containing characters to strip from the message text (XMBODY). It <i>must</i> be from 1 to 20 characters in length.
	("TDATE")	Add users to messages originating on or before this date. Must be any exact date recognized by VA FileMan. The default is the present. Used in conjunction with FLAGS.
	("TO PROMPT")	During interactive message addressing, contains the suggested initial addressee. The default is the user identified by XMDUZ.
	("TYPE")	Message type is one of the following special types: D—Document S—Spooled Document X—DIFROM O—ODIF B—BLOB K—KIDS
	("VAPOR")	Date/time (any format understood by VA FileMan) on which to delete (vaporize) this message from recipient baskets. Recipients can override this date.
	("WHEN")	Date/time (any format understood by VA FileMan) on which to print messages. The default is now.

Parameter	Description	Meaning
[.]xmto	Addressee or addressee array.	<p>If it is an array, it <i>must</i> be passed by reference.</p> <ul style="list-style-type: none"> - User's DUZ, or enough of user's name for a positive ID. For example: - 1301, "lastname,first", ARRAY(1301)="", or - ARRAY("lastname,first")="" - G.group name (enough for positive ID) - S.server name (enough for positive ID) - D.device name (enough for positive ID)
		<p>Prefix the above (except devices and servers) by:</p> <p>I:—For "Information Only" recipient (<i>cannot</i> reply). For example: "I:1301" <i>or</i> "I:lastname,first"</p> <p>C:—For "copy" recipient (not expected to reply). For example: "C:1301" <i>or</i> "C:lastname,first"</p> <p>L@datetime:—For when (in future) to send to this recipient (datetime can be anything accepted by VA FileMan). For example: "L@25 DEC@0500:1301" <i>or</i> "L@1 JAN:lastname,first" <i>or</i> "L@2981225.05:1301"</p> <p>(Can combine IL@datetime: or CL@datetime:)</p> <p>To delete recipients, prefix the recipients' name with a minus sign ("-"). For example: -1301 <i>or</i> "-lastname,first"</p> <p>To address any recipient (including users, groups, devices, and servers) at a remote site, just add the @site name. For example: recipient@site name</p>
xmk and xmkz for APIs that act on one message:		
xmk	(optional, depending on xmkz)	Basket (IEN or name) containing message.
xmkz	Identifies the message.	<p>Either:</p> <ul style="list-style-type: none"> - Message number (xmz) in the MESSAGE file (#3.9) (xmk must <i>not</i> be specified). - Message sequence number in the basket (xmk <i>must</i> be specified).
xmk and [.]xmkza for APIs that act on groups of messages:		
xmk	(optional, depending on xmkza)	Basket (IEN or name) containing messages.

Parameter	Description	Meaning
[.]xmkza	Identifies messages, using a list or list array, which can end in a comma.	Either: Message numbers (xmz) in the MESSAGE file (#3.9) (xmk must <i>not</i> be specified <i>and</i> ranges are <i>not</i> allowed): - List: "1234567" or "1234567,9763213" - List array: ARRAY(1234567)="" ARRAY(9763213)="" Message numbers in the basket (xmk <i>must</i> be specified and ranges <i>are</i> allowed): - List: "1" or "1,3,5-7" - List array: ARRAY("1,3")="" ARRAY("5-7")=""

4.2 ^XMXAPI

4.3 Message Actions—Building Block APIs

4.3.1 ADDRNSND^XMXAPI(): Address and Send Message

Reference Type	Supported
Category	Message Actions—Building Blocks
IA #	2729
Description	This API addresses and sends a message (does <i>not</i> handle the message body). It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



REF: Only the user or a surrogate can use this API.

Format	ADDRNSND^XMXAPI(xmduz,xmz,[.]xmto[.].xminstr]
Input Parameters	xmduz: (required) The user (DUZ or name) who is sending the message. xmz: (required) Message number in the MESSAGE file (#3.9). [.]xmto: (required) Recipients of the message. For a description of this parameter, please refer to the "Parameter Definitions" list above. Passed by reference or by value. .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: ADDR FLAGS", "FLAGS", "FROM", "LATER", "RCPT BSKT", "SCR HINT", "SCR KEY", "SELF BSKT", "SHARE BSKT", "SHARE DATE", "TYPE", "VAPOR"
Output	None

4.3.2 CRE8XMZ^XMXAPI(): Create a New Message Stub

Reference Type	Supported
Category	Message Actions—Building Blocks
IA #	2729

Description This API creates a new message stub in the MESSAGE file (#3.9). It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

Format CRE8XMZ^XMXAPI(xmsubj,.xmz)

Input Parameters xmsubj: (required) Subject of the message. It *must* be from 3 to 65 characters in length. If null, it defaults to "* No Subject *". If the subject is "* No Subject *" and the message is sent to a remote site, the subject in the "SUBJECT:" header record will be null.

Output Parameters .xmz: Message number in the MESSAGE file (#3.9).

 **REF:** See also GET^XMA2: Create Message Stub and XMZ^XMA2: Create Message Stub APIs described in Chapter 2, "Creating/Sending/Forwarding Messages," in this manual.

4.3.3 TOWHOM^XMXAPI(): Check One Message Addressee

Reference Type Supported

Category Message Actions—Building Blocks

IA # 2729

Description This API checks one message addressee. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

Format TOWHOM^XMXAPI(xmduz,xmz,xmtype,xmto[,xminstr],.xmfull)

Input Parameters xmduz: (required) The user (DUZ or name) who is addressing the message.
xmz: (required) Message number in the MESSAGE file (#3.9). This is not necessary if XMTYPE="S" and XMINSTR("ADDR FLAGS") contains "R".
xmtype: (required) Determines what prompts are used with the user:
S—User is sending a message.
F—User is forwarding a message.
xmto: (required) One recipient of the message. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual. This parameter must be passed by value.
.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "ADDR FLAGS"

Output Parameters .xmfull: Full name of the recipient.

 **REF:** See also: INST^XMA21: and WHO^XMA21 APIs described in Chapter 11, "Address Lookup," in this manual.


4.3.4 VSUBJ^XMXAPI(): Validate a Subject

Reference Type Supported


Category Message Actions—Building Blocks

IA # 2729

Description This API validates a subject. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

 **REF:** If the subject is "* No Subject *" and the message is sent to a remote site, the subject in the "SUBJECT:" header record will be null.


Format	VSUBJ^XMXAPI(.xmsubj)
Input Parameters	.xmsubj: (required) Subject of the message. It must be from 3 to 65 characters in length.
Output Parameters	.xmsubj: Subject of message. Leading and trailing blanks are removed, as are control characters. Any sequence of three or more consecutive blanks is reduced to two. If the subject is null, it defaults to "* No Subject *".

 **REF:** See also: \$\$SUBCHK^XMGAPI0(): Validate Message Subject API described in Chapter 2, "Creating/Sending/Forwarding Messages," in this manual.


4.4 Message Actions—APIs

4.4.1 ANSRMSG^XMXAPI(): Answer a Message

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API answers a message. (Send a new message, with a copy of the original message, to the sender of the original message). Sandwiches the user's answer (XMBODY) between the copy of the original message and a copy of the user's Network Signature. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.

 **REF:** Only the user or a surrogate with "WRITE" privilege can use this API. SHARED,MAIL cannot answer a message.

Format	ANSRMSG^XMXAPI(xmduz,xmk,xmkz[,xmsubj],xmbody[,xmtto][,xminstr],xmzr)
Input Parameters	<p>xmduz: (required) The user (DUZ or name) who is answering a message.</p> <p>xmk: (required) Message being answered. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.</p> <p>xmkz: (required) Message being answered. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.</p> <p>(optional) Subject of answer. It <i>must</i> be from 3 to 65 characters in length. If null, it defaults to: "Re: <subject of original message>"</p> <p>xmbody: (required) Closed root of array containing answer text. The root <i>cannot</i> be called "XMBODY". Also, it <i>must</i> be VA FileMan word-processing compatible.</p> <p>[.]xmtto: (optional) Additional recipients of answer. (Answer is automatically addressed to sender of original message.) Passed by reference or by value.</p> <p>.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" in this chapter: "ADDR FLAGS", "FLAGS", "FROM", "LATER", "RCPT BSKT", "SCR HINT", "SCR KEY", "SELF BSKT", "SHARE BSKT", "SHARE DATE", "STRIP", "TYPE", "VAPOR"</p>
Output Parameters	.xmzr: Message number (XMZ) of the answer in the MESSAGE file (#3.9).

 **REF:** See also: \$SENTA^XMA2R(): Create/Send Answer and Get Message Number API described in Chapter 6, "Replies/Answers to Messages—Creating and Sending," in this manual.

4.4.2 DELMSG^XMXAPI(): Delete Messages from a Basket

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API deletes messages from a basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



REF: Only the user or a surrogate can use this API.

Format	DELMSG^XMXAPI(xmduz,xmk,[.]xmkza,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose messages are to be deleted. xmk: (required) Messages to delete. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. [.]xmkza: (required) Messages to delete. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.
Output Parameters	.xmmsg: If deletion is completed successfully, contains the message: "<number of messages> deleted"



REF: See also: \$SENTA^XMA2R(): Create/Send Answer and Get Message Number API described in Chapter 6, "Replies/Answers to Messages—Creating and Sending," in this manual.

4.4.3 FLTRMSG^XMXAPI(): Filter Messages in a Basket

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API filters messages in a basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.




REF: Only the user or a surrogate can use this API.

Format	FLTRMSG^XMXAPI(xmduz,xmk,[.]xmkza,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose messages are to be filtered. xmk: (required) Messages to filter. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. [.]xmkza: (required) Messages to filter. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.
Output Parameters	.xmmsg: If filter is completed successfully, contains the message: "<number of messages> filtered"

4.4.4 FWDMSG^XMXAPI(): Forward Messages from a Basket

Reference Type	Supported
Category	Message Actions

IA # 2729
Description This API forwards messages from a basket. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.


 Only the user or a surrogate can use this API.

Format FWDMSG^XMXAPI(xmduz,xmk,[.]xmkza,[.]xmto[.,xminstr],.xmmsg)
Input Parameters
 xmduz: (required) The user (DUZ or name) whose messages are to be forwarded.
 xmk: (required) Messages to forward. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.
 [.]xmkza: (required) Messages to forward. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.
 [.]xmto: (required) To whom. For a description of this parameter, please refer to the "Parameter Definitions" list above. Passed by reference or by value.
 .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "ADDR FLAGS", "FWD BY", "LATER", "SELF BSKT", "SHARE BSKT", "SHARE DATE"
Output Parameters
 .xmmsg: If forward is completed successfully, contains the following message: "<number of messages> forwarded"

 **REF:** See also: ENT1^XMD: Forward a Message (Address Restrictions Waived) API described in Chapter 2, "Creating/Sending/Forwarding Messages," in this manual.

4.4.5 LATERMSG^XMXAPI(): "Later" Messages in a Basket


Reference Type Supported
Category Message Actions
IA # 2729
Description This API "Laters" messages in a basket. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.

 **REF:** Only the user or a surrogate can use this API.


Format LATERMSG^XMXAPI(xmduz,xmk,[.]xmkza[.,xminstr],.xmmsg)
Input Parameters
 xmduz: (required) The user (DUZ or name) whose messages are to be latered.
 xmk: (required) Messages to later. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.
 [.]xmkza: (required) Messages to later. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.
 .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "LATER"
Output Parameters
 .xmmsg: If later is completed successfully, contains the message: "<number of messages> latered"

4.4.6 MOVEMSG^XMXAPI(): Move Messages to Another Basket

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API moves messages from one basket to another basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.


 **REF:** Only the user or a surrogate can use this API.

Format	MOVEMSG^XMXAPI(xmduz,xmk,[.]xmkza,xmkto,.xmmsg)
Input Parameters	<p>xmduz: (required) The user (DUZ or name) whose messages are to be moved.</p> <p>xmk: (required) Messages to move. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.</p> <p>[.]xmkza: (required) Messages to move. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.</p> <p>xmkto: (required) Basket (IEN or name) to which to move the messages. The basket <i>must</i> already exist.</p>
Output Parameters	.xmmsg: If move is completed successfully, contains the message: "<number of messages> saved"

 **REF:** See also: S2^XMA1B: API described in Chapter 8, "Cross-category Activities—Mailboxes, Baskets, and Messages," in this manual.

4.4.7 PRTMSG^XMXAPI(): Print Messages

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API prints messages. (Actually, creates a task to print them.) It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

 **REF:** Only the user or a surrogate can use this API.

Format	PRTMSG^XMXAPI(xmduz,xmk,[.]xmkza,xmprtto[.,xminstr],.xmmsg,.xmtask[,xm subj][,xmto])
Input Parameters	<p>xmduz: (required) The user (DUZ or name) whose messages are to be printed.</p> <p>xmk: (required) Messages to print. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.</p> <p>[.]xmkza: (required) Messages to print. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.</p> <p>xmprtto: (required) Name of printer on which to print messages. This parameter is <i>not</i> checked, and <i>must</i> be correct.</p> <p>.xminstr: (optional) Appropriate special instructions For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "HDR", "RECIPS", "RESPTS", "WHEN"</p>

The following input parameters are only applicable if XMPRTTO is a P-MESSAGE device, and even then,

they are optional:

xmsubj: (optional) Subject of the P-MESSAGE message. The default is: "Queued Mail Report from <user name>."

Where <user name> is XMV("NAME").

.xmto: (optional) Additional recipients of P-MESSAGE message. (Message is automatically addressed to XMDUZ.)

Output Parameters

.xmmsg: If print is tasked successfully, contains the message, "<number of messages> printed."

.xmtask: If print is tasked successfully, contains the value of ZTSK.



REF: See also: PR2^XMA0: API described in Chapter 8, "Cross-category Activities—Mailboxes, Baskets, and Messages," in this manual.

4.4.8 PUTSERV^XMXAPI(): Put a Message in a Server Basket

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API puts a message in a server basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.
Format	PUTSERV^XMXAPI(xmkn,xmz)
Input Parameters	xmkn: (required) Full server name, including "S." xmz: (required) Message number in the MESSAGE file (#3.9).
Output	None



REF: See also: SETSB^XMA1C: API described in Chapter 15, "Servers—Message Activities," in this manual.

4.4.9 REPLYMSG^XMXAPI(): Reply to Message

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API replies to a message (Attach reply to original message). It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API.

Format	REPLYMSG^XMXAPI(xmduz,xmk,xmkz,xmbody[,..xminstr],.xmzr)
Input Parameters	xmduz: (required) The user (DUZ or name) who is replying to a message. xmk: (required) Message being replied to. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. xmkz: (required) Message being replied to. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. xmbody: (required) Closed root of array containing reply text. The root <i>cannot</i> be called "XMBODY". Also, it <i>must</i> be VA FileMan word-processing compatible.

.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "ADDR FLAGS", "FROM", "NET REPLY", "NET SUBJ", "SCR HINT", "SCR KEY", "STRIP"

Output Parameters .xmzr: Message number (XMZ) of the reply in the MESSAGE file (#3.9).



REF: See also: \$\$SENT^XMA2R(): Create/Send Reply and Get Message Number API described in Chapter 6, "Replies/Answers to Messages—Creating and Sending," in this manual.

4.4.10 SENDBULL^XMXAPI(): Send a Bulletin

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API sends a bulletin (returns XMZ). It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.
Format	SENDBULL^XMXAPI(xmduz,xmbname[,.xmparm][,xmbody][,.xmto][,.xminstr][,.xmattach])



NOTE: In addition to allowing a human user or a surrogate to call this API, MailMan Patch XM*8.0*36 allows an "Application Proxy," which is an application identifier that is a non-human user that does not have an Access code and/or mailbox, to call this API.

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Parameters	<p>xmduz: (required) The user (DUZ or name) who is sending the bulletin.</p> <p>xmbname: (required) Full name of the bulletin.</p> <p>.xmparm: (optional, unless parameters exist for the bulletin):</p>
XMPARM(<number>)=<value for parameter number>	<p>xmbody: (optional) Closed root of array containing additional bulletin text to append onto text predefined in bulletin. The root <i>cannot</i> be called "XMBODY". Also, it <i>must</i> be VA FileMan word-processing compatible.</p> <p>[.xmto: (optional) Additional recipients of the bulletin (in addition to those predefined in the bulletin). Passed by reference or by value.</p> <p>.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "ADDR FLAGS", "FLAGS", "FROM", "LATER", "RCPT BSKT", "STRIP", "TYPE", "VAPOR"</p> <p>.xmattach: (optional) Array of files to attach to the bulletin. (Reserved for future use.)</p>

Output Variables .XMZ: Message number (XMZ) of the bulletin in the MESSAGE file (#3.9).



REF: See also: EN^XMB: API described in Chapter 10, "Bulletins—Creating and Sending," in this manual.

4.4.11 SENDMSG^XMXAPI(): Send a Message

Reference Type Supported
Category Message Actions
IA # 2729
Description This API sends a message. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



REF: Only the user or a surrogate can use this API.

Format SENDMSG^XMXAPI(xmduz,xmsubj,xmbody,[.]xmtto[.],xmistr, XMZ[,xmattach])
Input Parameters xmduz: (required) The user (DUZ or name) who is sending the bulletin.
 xmsubj: (required) Subject of the message. It *must* be from 3 to 65 characters in length. If null, it defaults to "* No Subject *". If the subject is "* No Subject *" and the message is sent to a remote site, the subject in the "SUBJECT:" header record will be null.
 xmbody: (required) Closed root of array containing message text. The root *cannot* be called "XMBODY". Also, it *must* be VA FileMan word-processing compatible.
 [.]xmtto: (required) Recipients of the message. For a description of this parameter, please refer to the "Parameter Definitions" list above. Passed by reference or by value.
 .xmistr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "ADDR FLAGS", "FLAGS", "FROM", "LATER", "RCPT BSKT", "SCR HINT", "SCR KEY", "SELF BSKT", "SHARE BSKT", "SHARE DATE", "STRIP", "TYPE", "VAPOR"
 .xmattach: (optional) Array of files to attach to the message. (Reserved for future use.)
Output Parameters .xmz: Message number (XMZ) of the message in the MESSAGE file (#3.9); however, if \$D(XMINSTR("LATER")), then XMZ contains the task number of the task that will create the message at the specified later date.



REF: See also: ^XMD: Create and Send a Message API described in Chapter 2, "Creating/Sending/Forwarding Messages," in this manual.

4.4.12 TASKBULL^XMXAPI(): Send a Bulletin

Reference Type Supported
Category Message Actions
IA # 2729
Description This API sends a bulletin (quicker than SENDBULL, but does *not* return XMZ). It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



REF: Only the user or a surrogate can use this API.

Format	TASKBULL^XMXAPI(xmduz,xmbname[,xmparm][,xmbody][,xmtto][,xminstr],xm task[,xmattach])
Input Parameters	<p>xmduz: (required) The user (DUZ or name) who is sending the bulletin.</p> <p>xmbname: (required) Full name of the bulletin.</p> <p>.xmparm: (optional, unless parameters exist for the bulletin): XMPARM(<number>)=<value for parameter number></p> <p>xmbody: (optional) Closed root of array containing additional bulletin text to append onto text predefined in bulletin. The root <i>cannot</i> be called "XMBODY". Also, it <i>must</i> be VA FileMan word-processing compatible.</p> <p>[.xmtto: (optional) Additional recipients of bulletin (in addition to those predefined in bulletin). Passed by reference or by value.</p> <p>.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter: "ADDR FLAGS", "FLAGS", "FROM", "LATER", "RCPT BSKT", "STRIP", "TYPE", "VAPOR"</p> <p>.xmattach: (optional) Array of files to attach to the bulletin. (Reserved for future use.)</p>
Output Parameters	.xmtask: Task number (ZTSK) of the task that will create and send the bulletin.



REF: See also: ^XMB: API described in Chapter 10, "Bulletins—Creating and Sending," in this manual.

4.4.13 TERMMSG^XMXAPI(): Terminate Messages

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API terminates messages, possibly from a basket. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



REF: Only the user or a surrogate can use this API.

Format	TERMMSG^XMXAPI(xmduz,xmk,[.xmkza,xmmsg)
Input Parameters	<p>xmduz: (required) The user (DUZ or name) whose messages are to be terminated.</p> <p>xmk: (required) Messages to terminate. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.</p> <p>[.xmkza: (required) Messages to terminate. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value.</p>
Output Parameters	.xmmsg: If terminate is completed successfully, contains the message: "<number of messages> terminated"

4.4.14 VAPORMSG^XMXAPI(): Set Vaporize Date

Reference Type	Supported
-----------------------	-----------

Category	Message Actions
IA #	2729
Description	This API sets (schedules) a vaporize date/time for message(s) in one's basket(s) to be automatically deleted.
Format	VAPORMSG^XMXAPI(xmduz,xmk,[.]xmkza[.xminstr],xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose messages are to be vaporized. xmk: (required) Messages to vaporize. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. [.]xmkza: (required) Messages to vaporize. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter. XMKZA is passed by reference or by value. .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" list in this chapter.
Output Parameter	.xmmsg: If terminate is completed successfully, contains the message: "<number of messages> vaporized"

4.4.15 ZAPSERV^XMXAPI(): Delete a Message from a Server Basket

Reference Type	Supported
Category	Message Actions
IA #	2729
Description	This API deletes a message from a server basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.
Format	ZAPSERV^XMXAPI(xmkn,xmz)
Input Parameters	xmkn: (required) Full server name, including "S." xmz: (required) Message number in the MESSAGE file (#3.9)
Output	None



REF: See also: REMSBMSG^XMA1C: API described in Chapter 15, "Servers—Message Activities," in this manual.

5.0 Getting Information About and Text From Messages

5.1 ^XMAH

5.1.1 ENT8^XMAH: Display a List of All Responses to a Message (Interactive)

Reference Type	Supported
Category	Getting Information About and Text From Messages (Classic MailMan)
IA #	1040
Description	This API displays a list of all responses to a message interactively.
Format	ENT8^XMAH

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None

5.2 ^XMGAPI0

5.2.1 \$\$SUBGET^XMGAPI0(): Get Message Subject

Reference Type	Supported
Category	Getting Information About and Text From Messages (Classic MailMan)
IA #	1142
Description	This extrinsic function returns the message subject. Any ~U~ are automatically converted to a caret ("^"). If a message does not exist, it returns null.



REF: Compare this API to the \$\$SUBJ^XMXUTIL2(): and \$\$ZSUBJ^XMXUTIL2(): APIs described in Chapter 19, "Utilities—Message Information," in this manual.

Format	\$\$SUBGET^XMGAPI0(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Parameters returns: Returns message subject or null.



REF: See also: \$\$SUBJ^XMXUTIL2 API described in Chapter 19, "Utilities—Message Information," in this manual.

5.2.2 ^XMGAPI1

5.2.3 \$\$READ^XMGAPI1(): Get a Line of Text from a Message

Reference Type Supported
Category Getting Information About and Text From Messages (Classic MailMan)
IA # 1048
Description This extrinsic function returns a line of text from a message. By calling this API repeatedly, you can retrieve the lines of text, in order, from start to finish. The only thing this function does is: D REC^XMS3 Q XMRG Thus, it's just another way of invoking the REC^XMS3 API.



REF: Compare this API to the GET^XML: Retrieve Next Line of Message Text and REC^XMS3: Get a Line of Text from a Message APIs described in this chapter. For a description of the input and output variables, please refer to the REC^XMS3: Get a Line of Text from a Message API.



REF: If you want the whole text of a message, use VA FileMan's \$\$GET1^DIQ API.

Format \$\$READ^XMGAPI1
Input Parameters None
Output None

Example

```
S XMZ=message number in the MESSAGE file
F S LINE=$$READ^XMGAPI1() Q:XMER=-1 D
. ; line is in LINE, and also in XMRG
```

5.3 ^XMGAPI2

5.3.1 \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information

Reference Type Supported
Category Getting Information About and Text From Messages (Classic MailMan)
IA # 1144

Description This extrinsic function sets up (in ARRAY) an array of information about a message. It returns one of the following:

- If successful, it returns zero ("0").
- If not successful, then one of the following is returned:
 - "1-Undefined message number"
 - "1-No message number"
 - "1-No such message"
 - "2-User is not a sender of recipient."
 - "4-Invalid user"



REF: Compare this API to the `$$NET^XMRENT()`: Get Message Information API described in this chapter and the `INMSG^XMXUTIL2()`: Get Message Information and all other APIs in `^XMXUTIL2` described in Chapter 19, "Utilities—Message Information," in this manual.

Format `$$HDR^XMGAPI2(xmz,.array,flag)`

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	Description	Meaning
DUZ	(required) For a description of this variable, please refer to Table 1 1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.	
XMDUZ	(optional) DUZ of the user. The default is DUZ.	
XMZ	(required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).	
	Flag	(required) Flag that determines what message information is placed in the output ARRAY parameter: <ul style="list-style-type: none"> - 0 or Undefined – Return basic information - 1 – return basic information + response and BLOB count information - 91 – return flag 1 information + response IDs. - 92 – return flag 1 information + BLOB IDs. - 93 – return all of the above.

Output Parameter	Description	Meaning
array	Message information array	
	If FLAG =or undefined	
	("BROADCAST")	1—If the message was broadcast. 0—If the message was not broadcast.
	("BSKT")	Basket name (XMDUZ); null if not in basket.
	("BSKT IEN")	Basket IEN (XMDUZ); null if not in basket.
	("DATE")	Message date/time, in format MAY 25, 1999@08:16:00, if local, or as sent, if remote.
	("DATE FM")	Message date/time (VA FileMan format); date only if remote.
	("LINES")	Number of lines in the original message.
	("NEW")	= 1 if the message is new; 0 otherwise.
	("PXMZ")	Message number of original message; null if not a response.
	("SENDER")	Sender (from, external format).
	("SENDER DUZ")	Sender (from) DUZ; null if remote or fictitious.
	("SUBJ")	Subject (external format).
	("SURROG")	Surrogate sender (external format).
	("TYPE")	Message type (piece 7 of the message's zero node).
	("XMZ")	Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
		1 = returns function value and value array above, also additional value array as follows:
	("RRED")	Responses read (XMDUZ); null if not applicable.
	("RRCV")	Responses received; null if not applicable.
	("BLOBCNT")	Number of non-textual body parts attached (for the Imaging software).
	If FLAG = 91 and if the message has responses, returns value array as with Flag 1 and an array of response nodes and values as follows:	
	("RSP",i)	(Pointer to the MESSAGE file [#3.9]) array of responses.
	If FLAG = 92 and if the message has BLOBS, it returns the value array as with Flag 1 and an array of non-textual body parts as follows:	
	("BLOB",i)	(Pointer to the OBJECT file [#2005]) array of BLOBS (for the Imaging software)
	If FLAG = 93, it returns all of the above.	



REF: See also: INMSG^XMXUTIL2(): Get Message Information, INMSG1^XMXUTIL2(): Get Message Information (Part 1), INMSG2^XMXUTIL2(): Get Message Information (Part 2), INRESPTS^XMXUTIL2 APIs described in Chapter 19, "Utilities—Message Information," in this manual.

5.3.2 ^XML

5.3.2.1 GET^XML: Retrieve Next Line of Message Text

Reference Type	Supported
Category	Getting Information About and Text From Messages (Classic MailMan)
IA #	1283
Description	This API sets up an executable variable (XMREC), which, when executed, retrieves the next line of text from a message. Thus, by executing XMREC repeatedly, you can retrieve the lines of text, in order, from start to finish. This call creates several variables, some of which are documented here for informational purposes only.



REF: Compare this API to the \$\$READ^XMGAPI1(): Get a Line of Text from a Message and REC^XMS3: Get a Line of Text from a Message APIs described in this chapter.

Format GET^XML

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMCHAN: (required) Must be set to "SERVER", or another protocol defined in the COMMUNICATIONS PROTOCOL file (#3.4).
Output Variables	XMCHAN: IEN, in the COMMUNICATIONS PROTOCOL file (#3.4), of the Input XMCHAN. XMCLOSE: (Ignore) This is the variable, which, when executed, closes the communications channel. XMOPEN: (Ignore) This is the variable, which, when executed, opens the communications channel. XMPROT:: Copy of Input XMCHAN. XMREC: This is the variable, which, when executed, retrieves the next line of text of a message. XMSEN: (Ignore) This is the variable, which, when executed, sends the next line of text of a message.

Example

```
>S XMCHAN="SERVER" D GET^XML
```

The XMREC variable, which, when executed, does, the following:

```
>D REC^XMS3
```

Thus, it's another way of invoking REC^XMS3. Continuing:

```
S XMZ=message number in the MESSAGE file
F X XMREC Q:XMER=-1 D
. ; line is in XMRG
```



REF: For a description of the input and output variables, please refer to the REC^XMS3: Get a Line of Text from a Message API in this chapter.

5.4 ^XMRENT

5.4.1 \$\$NET^XMRENT(): Get Message Information

Reference Type	Supported
Category	Getting Information About and Text From Messages (Classic MailMan)
IA #	1143
Description	This extrinsic function returns an ^-delimited string of information about a message. If message does <i>not</i> exist, returns null.



REF: Compare this API to the \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API described in this chapter and the INMSG^XMXUTIL2(): Get Message Information API and all other APIs in ^XMXUTIL2 described in Chapter 19, "Utilities—Message Information," in this manual.

Format	\$\$NET^XMRENT(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	Return. Returns: Piece 1: Message date, in the following format: MAY 25, 1999@08:16:00, if local, or as sent, if remote. Piece 2: Scramble hint, if any; otherwise null. Piece 3: Message from (external). Piece 4: Message ID at originating site (XMZ@sitename, if local). Piece 5: Message sender, usually surrogate (external). Piece 6: Message subject (external). Piece 7: Message ID of original message, if this is a reply (XMZ@sitename, if local). Piece 8: Message type (Piece 7 of the message's zero node).

5.5 ^XMS3

5.5.1 REC^XMS3: Get a Line of Text from a Message

Reference Type	Supported
Category	Getting Information About and Text From Messages (Classic MailMan)
IA #	10073
Description	This API gets a line of text from a message. By calling this API repeatedly, you can retrieve the lines of text, in order, from start to finish.



REF: Compare this API to the `$$READ^XMGAPI1()`: Get a Line of Text from a Message and `GET^XML`: Retrieve Next Line of Message Text APIs described in this chapter.

Format REC^XMS3

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	<p>XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>XMPOS: (optional) Line number of previous line read. The default is .99. Thus, the first time this API is called, XMPOS should not be defined, unless you want to start reading the message at some line other than line 1. If it's a message from a remote site, and you want to read the header records, set XMPOS=0. For every subsequent call to this API, XMPOS is already set to read the next line, so you do not need to set it.</p>
Output Variables	<p>XMER: End of text reached? 0—No -1—Yes</p> <p>XMPOS: Line number of the latest line read. (null, if XMER=-1)</p> <p>XMRG: Text of latest line read. (null, if XMER=-1)</p>

Example

```
S XMZ=message number in the MESSAGE file
F D REC^XMS3 Q:XMER=-1 D
. ; line is in XMRG
```


6.0 Replies/Answers to Messages—Creating and Sending

6.1 ^XMA11A

6.1.1 WRITE^XMA11A: Answer a Message (Interactive)

Reference Type	Supported
Category	Replies/Answers to Messages—Creating and Sending (Classic MailMan)
IA #	1233
Description	This API answers a message interactively.
Format	WRITE^XMA11A

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. X: (required) Must be set to "A", otherwise this call sends a new message. XMDUZ: (optional) User's DUZ. XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9), of the message to which you are sending an answer.
Output Variables	None

6.2 ^XMA2R

6.2.1 \$\$ENT^XMA2R(): Create/Send Reply and Get Message Number

Reference Type	Supported
Category	Replies/Answers to Messages—Creating and Sending (Classic MailMan)
IA #	1145
Description	This extrinsic function creates and sends a reply to a message and returns the message number of the reply. If the reply is not successful, returns a string with the text of the error. Unlike an answer, a reply is sent to all (local) recipients of the message to which you are replying.



REF: Compare this API to the `$$ENTA^XMA2R()`: API described in this chapter and the `ANSRMSG^XMXAPI()`: Answer a Message and `REPLYMSG^XMXAPI()`: Reply to Message APIs described in Chapter 4, "Message Actions," in this manual.

Format `$$ENTA^XMA2R(xmz,xmsub,.xmreply[,xmstrip][,xmnet])`

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p>
Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9), of the message to which you are replying.</p> <p>xmsub: (required) Subject of the reply (ignored, unless message is from a remote sender).</p> <p>.xmreply: (required) Text of the reply. Must be in a local array passed by reference. It must be in a format acceptable to VA FileMan word-processing fields.</p> <p>xmstrip: (optional) String containing characters that should be removed from the reply text. The default is none.</p> <p>xmnet: (optional) If the sender of the original message is at a remote site, should the reply be sent to the sender, too? (Ignored, unless message is from a remote sender.) 0 (default)—No 1—Yes</p>
Output	<p>returns: Returns: Successful—Message number of the reply. Unsuccessful—A string with the text of the error.</p>

6.2.2 `$$ENTA^XMA2R()`: Create/Send Answer and Get Message Number

Reference Type	Supported
Category	Replies/Answers to Messages—Creating and Sending (Classic MailMan)
IA #	1145
Description	This extrinsic function creates and sends an answer to a message and returns the message number of the answer. If the answer is not successful, it returns a string with the text of the error. Unlike a reply, an answer is sent only to the sender of the original message. It makes no difference whether the sender is local or remote.



REF: Compare this API to the `$$ENT^XMA2R()`: API described in this chapter and the `ANSRMSG^XMXAPI()`: Answer a Message and `REPLYMSG^XMXAPI()`: Reply to Message APIs described in Chapter 4, "Message Actions," in this manual.

Format `$$ENTA^XMA2R(xmz,xmsub,xmtext[,xmstrip])`

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9), of the message to which you are sending an answer. xmsub: (required) Subject of the answer. xmtext: (required) Text of the answer. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. xmstrip: (optional) String containing characters that should be removed from the answer text. The default is none.
Output	returns: Returns: Successful—Message number of the answer Unsuccessful—A string with the text of the error.

6.3 ^XMAH1

6.3.1 ^XMAH1: Create/Send a Reply to a Message (Interactive)

Reference Type	Supported
Category	Replies/Answers to Messages—Creating and Sending (Classic MailMan)
IA #	1232
Description	This API creates and sends a reply to a message interactively.
Format	<code>^XMAH1</code>

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMK: (required) IEN of the user's basket in which the message resides.</p> <p>XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9), of the message to which you are replying.</p>
Input Variables	XMDF: (optional) If \$D(XMDF) all addressing restrictions are waived.
Output Variables	None

6.3.1.1 ENTA^XMAH1: Create/Send a Reply to a Message (Interactive)

Reference Type	Supported
Category	Replies/Answers to Messages—Creating and Sending (Classic MailMan)
IA #	1232
Description	This API creates and sends a reply to a message interactively.



REF: This API is identical to the ^XMAH1: Create/Send a Reply to a Message (Interactive) API.

Format	ENTA^XMAH1
Core Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMK: (required) IEN of the user's basket in which the message resides.</p> <p>XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9), of the message to which you are replying.</p>
Input Variables	XMDF: (optional) If \$D(XMDF) all addressing restrictions are waived.
Output Variables	None

7.0 Basket Actions

7.1 ^XMA03

7.1.1 \$\$REN^XMA03: Perform Integrity Check on User Basket

Reference Type	Supported
Category	Basket Actions
IA #	1150
Description	This extrinsic function performs an integrity check on a user's basket, resequences messages in a user's basket, and returns a string, "Resequenced from 1 to n", where n is the number of messages in the basket.
Format	\$\$REN^XMA03



REF: Compare this API to the RSEQBSKT^XMXAPIB(): API described in this chapter.

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (required) DUZ of user. XMK: (required) Basket Internal Entry Number (IEN).
Output Variables	returns: Returns a string, "Resequenced from 1 to n", where n is the number of messages in the basket.

7.2 ^XMAD2

7.2.1 \$\$BSKT^XMAD2: Basket Lookup

Reference Type	Supported
Category	Basket Actions
IA #	1147
Description	This extrinsic function, given a basket name and a user's DUZ, looks up the basket. If it does not exist, creates it and returns its Internal Entry Number (IEN). If it does exist, return its IEN. If there's an error, it returns an error message.

Format \$\$BSKT^XMAD2

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables XMDUZ: (required) DUZ of user.

XMKN: (required) Basket name.

Output

returns: Returns:

Successful—Basket IEN.

Unsuccessful—Error message.

7.3 ^XMXAPIB

7.3.1 CRE8BSKT^XMXAPIB(): Create a Basket

Reference Type Supported

Category Basket Actions

IA # 2723

Description This API creates a basket. If the basket already exists, an error is returned. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API. If the user is SHARED,MAIL, then the surrogate must be a Postmaster surrogate or hold the XMMGR security key.

Format CRE8BSKT^XMXAPIB(xmduz,xmkn,.xmk)

Input Parameters xmduz: (required) The user (DUZ or name) for whom a basket is to be created.

xmkn: (required) The name of the basket. Basket name is FREE TEXT. It can be from 2 to 30 characters in length.

Output Parameters .xmk: Basket number. The Internal Entry Number IEN of the basket created.

returns: Returns:

Successful—Creates Basket.

Unsuccessful—Error message.

7.3.2 CRE8MBOX^XMXAPIB(): Create a Mailbox

Reference Type Supported

Category	Basket Actions
IA #	2723
Description	This API creates a mailbox for a user. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.
Format	CRE8MBOX^XMXAPIB(xmduz[,xmdate])
Input Parameters	xmduz: (required) The user (DUZ or name) for whom the mailbox is to be created. xmdate: (optional) Users who are being reinstated after not having worked at the VA for a while can be restricted from seeing messages earlier than a certain date. If the user is a first-time user, then this parameter has no effect and should not be used. 0 or Null—The user can access any message on the system that was ever addressed to the user. Date—The user <i>cannot</i> access any message addressed to the user on the system earlier than this date unless it is already in the user's mailbox or if someone forwards it to the user.
Output	None

7.3.3 DELBSKT^XMXAPIB(): Delete a User's Basket

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API deletes a user's basket. The special baskets ("IN" and "WASTE") <i>cannot</i> be deleted. Only empty baskets can be deleted, unless XMFLAGS contains "D." It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API. If the user is SHARED,MAIL, then the surrogate must be a Postmaster surrogate or hold the XMMGR security key.

Format	DELBSKT^XMXAPIB(xmduz,xmk[,xmflags])
Input Parameters	xmduz: (required) The user (DUZ or name) for whom a basket is to be deleted. xmk: (required) Basket (IEN or name) to be deleted. xmflags: (optional) Used to control processing: D—Delete the basket even if there are messages in it.
Output	None

7.3.4 FLTRBSKT^XMXAPIB(): Filter Messages in a Basket

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API filters messages in a basket. It runs all messages in a basket through any filters that may exist for the mailbox. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API. If the user is SHARED,MAIL, then the surrogate must be a Postmaster surrogate or hold the XMMGR security key.

Format	FLTRBSKT^XMXAPIB(xmduz,xmk,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose basket is to be filtered. xmk: (required) Basket (IEN or name) to be filtered.

Output Parameters .xmmsg: If filtering is completed successfully, this parameter contains the message, "Basket filtered."

7.3.5 FLTRMBOX^XMXAPIB(): Filter All Messages in a User's Mailbox

Reference Type Supported
Category Basket Actions
IA # 2723
Description This API filters all messages in a user's mailbox. Runs all messages in all baskets in the user's mailbox through any filters that may exist for the mailbox. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API

Format FLTRMBOX^XMXAPIB(xmduz,.xmmsg)
Input Parameters xmduz: (required) The user (DUZ or name) whose mailbox (all baskets) is to be filtered.
Output Parameters .xmmsg: If filtering is completed successfully, contains the message, "Mailbox filtered".

7.3.6 LISTBSKT^XMXAPIB(): Get a List of Baskets in a Mailbox

Reference Type Supported
Category Basket Actions
IA # 2723
Description This API gets a list of baskets in a mailbox. Gets a list (similar in format to that produced by LIST^DIC) of a user's baskets, optionally restricting the list to only those baskets with new mail, and/or those baskets whose name starts with a certain string. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Regardless of the alphabetic order you request, lowercase names sort separately from uppercase names; therefore, an all uppercase cross-reference (under "BSKT") is provided, if you do not limit the number of entries returned.



NOTE: Only the user or a surrogate can use this API.

Format LISTBSKT^XMXAPIB(xmduz[,xmflags][,xmamt][,.xmstart][,xmpart],xmtree)
Input Parameters xmduz: (required) The user (DUZ or name) for whom a basket list is to be compiled.
xmflags: (optional) Used to control processing:
B—Backwards alpha order (the default is alpha order)
N—Baskets with new messages only
xmamt: (optional) How many? * (default)—Get all and provide uppercase basket name cross-reference
Number—Get this many
.xmstart: (optional) Used to start the Lister going. The Lister will keep it updated from call to call.
xmpart: (optional) List only those baskets whose name starts with this string.

Output

xmtrout: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).
None

Example

In the following example, notice that:

- The node "XMLIST" has been added under the target root that was specified.
- The header node of the list is identical to the header node produced by VA FileMan's LIST^DIC.
- The list is in alphabetical order. Unlike VA FileMan's LIST^DIC, the counter starts at 1, whether you've requested forward or reverse order, and when you traverse the list, starting at 1, you are traversing in the order you requested.
- The data for each basket is the same as that produced by QBSKT^XMXAPIB, namely:

Piece 1: Basket IEN.
Piece 2: Basket name.
Piece 3: Number of messages in the basket.
Piece 4: Number of new messages in the basket.

```
>K XMSTART
>D LISTBSKT^XMXAPIB(3,,6,.XMSTART,, "TEST" )
>ZW TEST
TEST("XMLIST",0)=6^6^1
TEST("XMLIST",1)=10^98 BASKET^1^
TEST("XMLIST",2)=8^DELIVERY^1^
TEST("XMLIST",3)=2^FOUR^4^0
TEST("XMLIST",4)=9^FORTYFOUR^0^
TEST("XMLIST",5)=1^IN^335^1
TEST("XMLIST",6)=4^FIVE^10^0

>ZW XMSTART
XMSTART=FIVE
XMSTART(" IEN" )=4

>D LISTBSKT^XMXAPIB(3,,5,.XMSTART,, "TEST" )
>ZW TEST
TEST("XMLIST",0)=5^5^0
TEST("XMLIST",1)=3^SIX^9^0
TEST("XMLIST",2)=6^SEVEN^13^0
TEST("XMLIST",3)=5^EIGHT^4^0
TEST("XMLIST",4)=7^SCRAMBLE^3^0
TEST("XMLIST",5)=.5^WASTE^44^

>ZW XMSTART
XMSTART=
XMSTART(" IEN" )=
```

7.3.7 LISTMSGSEXAPIB(): Get a List of Messages in a Mailbox

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API gets a list of messages in a mailbox. It gets a list (similar in format to that produced by LIST^DIC) of messages in one basket or all baskets, optionally based on certain criteria. The IENs of the messages in the MESSAGE file (#3.9) are returned. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API.

Format	LISTMSGSEXAPIB(xmduz,xmk[,xmflds][,xmflags][,xamt][,xmstart][,xmcrit][,xmtroot])
Output	None.

The **Input Parameters** are shown in the following table:

Input Parameters	Description	Meaning
Xmduz	(required)	The user (DUZ or name) for whom a message list is to be compiled.
Xmk	(required)	Basket in which to look: <ul style="list-style-type: none"> • IEN or Name—Look in this basket only • *—Look in all baskets
xmflds	(optional)	A string containing a list of fields to retrieve, separated by ";". The default is none. For example, XMFLDS="SUBJ;DATE" means retrieve subject and date.
	"BSKT"	Basket (default: <basket IEN>^<basket name>) Optionally followed by ":" and "I" for basket IEN only (no 2nd piece). "X" adds basket name cross-reference.
	"DATE"	Date sent (default: <internal date>^<dd mmm yy hh:mm>). Optionally followed by ":" and "I" for internal only (no 2nd piece). "F" for VA FileMan date as the 2nd piece. "X" adds VA FileMan date cross-reference.
	"FROM"	Message from (default: <internal from>^<external from>) Optionally followed by ":" and "I" for internal only (no 2nd piece) "X" adds external "From" cross-reference.
	"LINE"	Number of lines in the message.

Input Parameters	Description	Meaning
	"NEW"	Is the message new? <ul style="list-style-type: none"> • 0—No • 1—Yes • 2—Yes, and priority too
	"PRI"	Is the message priority? <ul style="list-style-type: none"> • 0—No • 1—Yes
	"READ"	How much of the message has the user read? <ul style="list-style-type: none"> • Null—Has not read the message at all • 0—Has read the message, but no responses • Number—Has read through this response
	"RESP"	How many responses does the message have? <ul style="list-style-type: none"> • 0—None • Number—This many
	"SEQN"	Sequence number in basket.
	"SUBJ"	Message subject (always external). Optionally followed by ":" and "X" adds message subject cross-reference.
xmflags	(optional)	Used to control processing: <ul style="list-style-type: none"> • B—Backwards order (the default is traverse forward) • C—Use basket C-cross-reference (the default is message IEN) • N—New messages only (C flag ignored) • P—New Priority messages only (C, N flags ignored)
xmamt	(optional)	How Many? <ul style="list-style-type: none"> • * (default)—Get all • Number—Get this many
.xmstart	(optional)	Used to start the Lister going. The Lister will keep it updated from call to call: ("XMK") Start with this basket IEN (valid only if XMK="*"). Continues from there, with each successive call, to the end. The default is to start with basket ".5"—the "WASTE" basket. ("XMZ") Start after this message IEN (valid only if no C flag). Continues from there, with each successive call, to the end. The default is to start at the beginning (or end) of the basket. ("XMKZ") Start after this message sequence number (valid only if C flag). Continues from there, with each successive call, to the end. The default is to start at the beginning [or end] of the basket.

Input Parameters	Description	Meaning
.xmcrit	(optional)	Criteria that are put together using an "and" statement to select only those messages that meet the criteria. The default is to get a list of all messages in the basket or mailbox.
	("FROM")	Message is from this person. If the person is a local user, it must be the user's DUZ. If the person is a remote user, it must be <partial name>_ "@ " _<partial domain>. The search is not case sensitive. For example: "four@"—Finds any person whose name contains "four". "@va"—Finds any person whose domain contains "va". "four@va"—Finds any person whose name contains "four" and domain contains "va".
	("FDATE")	Message was sent on or after this date. Date <i>must</i> be in VA FileMan format.
	("RFROM")	Message has a response from this person (same rules as "FROM").
	("SUBJ")	Subject contains this string.
	("SUBJ","C")	Is search case sensitive? <ul style="list-style-type: none"> • 0 (default)—Search is not case sensitive • 1—Search is case sensitive
	("TDATE")	Message was sent on or before this date. Date <i>must</i> be in VA FileMan format.
	("TEXT")	Message contains this string.
	("TEXT","L")	Where should we look for the text? <ul style="list-style-type: none"> • 1 (default)—Look in message only • 2—Look in both message and responses • 3—Look in responses only
	("TEXT","C")	Is search case sensitive? <ul style="list-style-type: none"> • 0 (default)—Search is not case sensitive • 1—Search is case sensitive
	("TO")	Message is to this person (same rules as "FROM").
	XMTROOT	Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).

Example

```
>D LISTMSGs^XMXAPIB(3,2,"FROM:X;SUBJ;DATE:IX","B")
```

In the following example, notice that:

- The list is in reverse XMZ order (because of the "B" flag). Unlike VA FileMan's LIST^DIC, the counter starts at one, whether you've requested forward or reverse order, and when you traverse the list, starting at one, you are traversing in the order you requested.
- The header node of the list is identical to the header node produced by VA FileMan's LIST^DIC.
- The cross-reference for the date field is adjusted to local time (PST, in this case) if the message is from a remote site.
- The external form of the from field removes the "<>" if the message is from a remote site.

```

^TMP("XMLIST",564222283,0) = 4^^^0
^TMP("XMLIST",564222283,1) = 978181
^TMP("XMLIST",564222283,1,"DATE") = 08 Apr 97 15:08 PST
^TMP("XMLIST",564222283,1,"FROM") = <XMUSER.ONE_M+@ISC-SF.VA.GOV>^
XMUSER.ONE_M+@ISC-SF.VA.GOV
^TMP("XMLIST",564222283,1,"SUBJ") = FOUR
^TMP("XMLIST",564222283,2) = 977961
^TMP("XMLIST",564222283,2,"DATE") = 02 Dec 96 15:37 EST
^TMP("XMLIST",564222283,2,"FROM") = <XMUSER.THREE@FORUM.VA.GOV>^
XMUSER.THREE@FORUM.VA.GOV
^TMP("XMLIST",564222283,2,"SUBJ") = XMTHREE UTILITY
^TMP("XMLIST",564222283,3) = 977827
^TMP("XMLIST",564222283,3,"DATE") = 2960809.10363
^TMP("XMLIST",564222283,3,"FROM") = 3^XMUSER,ONE
^TMP("XMLIST",564222283,3,"SUBJ") = TEST NEW PROTOCOL
^TMP("XMLIST",564222283,4) = 977647
^TMP("XMLIST",564222283,4,"DATE") = 2960725.142942
^TMP("XMLIST",564222283,4,"FROM") = .5^POSTMASTER
^TMP("XMLIST",564222283,4,"SUBJ") = G.FOUR@ISC-SF.VA.GOV NOT FOUND
^TMP("XMLIST",564222283,"DATE",2960725.142942,4) =
^TMP("XMLIST",564222283,"DATE",2960809.10363,3) =
^TMP("XMLIST",564222283,"DATE",2961202.1337,2) =
^TMP("XMLIST",564222283,"DATE",2970408.1508,1) =
^TMP("XMLIST",564222283,"FROM", "XMUSER,ONE", 3) =
^TMP("XMLIST",564222283,"FROM", "XMUSER,ONE_M+@ISC-SF.VA.GOV", 1) =
^TMP("XMLIST",564222283,"FROM", "POSTMASTER", 4) =
^TMP("XMLIST",564222283,"FROM", "XMUSER.THREE@FORUM.VA.GOV", 2) =

```

7.3.8 NAMEBSKT^XMXAPIB(): Change the Name of a Basket

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API changes the name of a basket. The "IN" and "WASTE" baskets <i>cannot</i> be renamed. It sets XMERR and ^TMP("XMERR", \$J), if an error occurs.



NOTE: Only the user or a surrogate with "WRITE" privileges can use this API. If the user is SHARED,MAIL, then the surrogate must be a Postmaster surrogate or hold the XMMGR security key.

Format	NAMEBSKT^XMXAPIB(xmduz,xmk,xmkn)
Input Parameters	xmduz: (required) The user (DUZ or name) whose basket is to be renamed. xmk: (required) Basket (IEN or name) to be renamed. xmkn: (required) The new name of the basket. Basket name is FREE TEXT. It can be from 2 to 30 characters in length.
Output	None

7.3.9 QBSKT^XMXAPIB(): Get information on a basket

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API gets information on a basket. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.



NOTE: Only the user or a surrogate can use this API.

Format	QBSKT^XMXAPIB(xmduz,xmk,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose basket is to be queried. xmk: (required) Basket (IEN or name) to be queried.
Output Parameters	.xmmsg: String containing the following pieces of information, separated by a caret (^): Piece 1: Basket IEN. Piece 2: Basket name. Piece 3: Number of messages in the basket. Piece 4: Number of new messages in the basket.

7.3.10 QMBOX^XMXAPIB(): Query a Mailbox for New Messages

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API queries a mailbox for new messages. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.




NOTE: Only the user or a surrogate can use this API.

Format	QMBOX^XMXAPIB(xmduz,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose mailbox is to be queried.
Output Parameters	.xmmsg: If no new messages, string is 0. If there are new messages, string contains the following pieces of information, separated by a caret (^): Piece 1: Number of new messages in the mailbox. Piece 2: Does the user have new priority mail? 0—No 1—Yes Piece 3: Number of new messages in the "IN" basket. Piece 4: Date/time (in VA FileMan format) that the last message was received. Piece 5: Have there been any new messages since the last time this routine was called: 0—No 1—Yes

7.3.11 RSEQBSKT^XMXAPIB(): Resequence Messages in a Basket


Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API resequences messages in a basket. Before the resequencing is done, a basket integrity check is performed, and any errors detected are corrected. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

 **NOTE:** Only the user or a surrogate can use this API. If the user is SHARED,MAIL, then the surrogate must be a Postmaster surrogate or hold the XMMGR security key.

Format	RSEQBSKT^XMXAPIB(xmduz,xmk,.xmmsg)
Input Parameters	xmduz: (required) The user (DUZ or name) whose basket is to be resequenced. xmk: (required) Basket (IEN or name) to be resequenced.
Output Parameters	.xmmsg: If resequencing is completed successfully, contains the message: "Resequenced from 1 to <number of messages in basket>"

7.3.12 TERMMBOX^XMXAPIB(): Remove All Traces of a User from MailMan Globals

Reference Type	Supported
Category	Basket Actions
IA #	2723
Description	This API Remove all traces of a user from MailMan globals. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.

 **NOTE:** Only a Postmaster surrogate or users who hold the XMMGR security key can use this API.

Format	TERMMBOX^XMXAPIB(xmduz)
Input Parameters	xmduz: (required) The user (DUZ or name) whose mailbox is to be terminated.
Output	None

8.0 Cross-category Activities—Mailboxes, Baskets, and Messages

8.1 ^XM

8.1.1 \$\$NU^XM: Get Number of New Messages

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	10064
Description	This extrinsic function returns the number of new messages the user has, and it may display a message to the user telling how many.



NOTE: Compare this API to the QMBOX^XM/APIB(): Query a Mailbox for New Messages and QBSKT^XM/APIB(): Get information on a basket APIs described in Chapter 7, "Basket Actions"; the \$\$BNMSGCT^XM/UTIL(): , \$\$TNMSGCT^XM/UTIL(): , and \$\$NEWS^XM/UTIL(): APIs described in Chapter 17, "Utilities—Messages and Mailboxes."

Format \$\$NU^XM(force)

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) DUZ of user. XMDUZ: (optional) DUZ of user. The default is DUZ.
Input Parameters	force: (required) Force MailMan to display a message to the user, telling how many new messages the user has? "You have <x> new messages." 0—No, only display it if the user has received new messages since the last time the user was told 1—Yes, display it
Output	None

8.1.2 ^XMA

8.1.2.1 REC^XMA: Read/Manage Messages (Interactive)

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	1284
Description	This API reads/manages messages interactively. It is identical to the Read/Manage Messages option [XMREAD]. This API call can be placed in a menu option as follows: Entry action: S XMMENU(0)=<name of the menu option> Routine: REC^XMA Exit action: K XMMENU D CHECKOUT^XM



REF: Compare this API to the READ^XMXAPIU API described in Chapter 13, "User Actions—Interactive," in this manual.

Format REC^XMA

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

Output Variables None

8.1.3 ^XMA0

8.1.3.1 ENTPRT^XMA0: Print a Message (Interactive)

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	1230
Description	This API prints a message interactively.
Format	ENTPRT^XMA0

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.

- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. XMDUZ: (optional) DUZ of the user. The default is DUZ. XMK: (required) Basket IEN. XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None

8.1.3.2 HDR^XMA0: Headerless Print a Message

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	1230
Description	This API prints a message <i>without</i> a header.



REF: Compare this API to the PR2^XMA0: API described in this chapter and the PRTMSG^XMXAPI(): Print Messages API described in Chapter 4, "Message Actions," in this manual.

Format	HDR^XMA0
---------------	----------

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. IO: (required) Device to which to print. XMDUZ: (optional) DUZ of the user. The default is DUZ. XMK: (required) Basket IEN. XMTYPE: (optional) If not defined, the message is printed in its entirety. Otherwise, ";"-piece 6 is the number of the response from which to print. (If ";"-piece 6 is null or zero, then the message is printed in its entirety.) If XMTYPE="^", then this API aborts. XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None

8.1.3.3 PR2^XMA0: Print a Message

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	1230
Description	This API prints a message.



REF: Compare this API to the HDR^XMA0: described in this chapter and the PRTMSG^XMXAPI(): Print Messages API described in Chapter 4, "Message Actions," in this manual.

Format PR2^XMA0

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
IO: (required) Device to which to print.
XMDUZ: (optional) DUZ of the user. The default is DUZ.
XMK: (required) Basket IEN.
XMTYPE: (optional) If not defined, the message is printed in its entirety. Otherwise, ";"-piece 6 is the number of the response from which to print. (If ";"-piece 6 is null or zero, then the message is printed in its entirety.) If XMTYPE="^", then this API aborts.
XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Variables None

8.1.4 ^XMA1B

8.1.4.1 KL^XMA1B: Delete a Message from a Basket

Reference Type	Supported
Category	Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA #	10065
Description	This API deletes a message from a basket. Unlike the KLQ^XMA1B: API, the message is <i>not</i> put in the "WASTE" basket.



REF: Compare this API to the KLQ^XMA1B: API described in this chapter and the DELMSG^XMXAPI(): Delete Messages from a Basket API described in Chapter 4, "Message Actions," in this manual.

Format KL^XMA1B

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables XMDUZ: (required) DUZ of the user who owns the basket.
 XMK: (optional) Basket IEN. If \$G(XMK), then MailMan looks to see in which basket the message is located.
 XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Variables None

8.1.4.2 KLQ^XMA1B: Delete a Message from a Basket (into "WASTE" basket)

Reference Type Supported
Category Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)
IA # 10065
Description This API Delete a message from a basket and put it in the "WASTE" basket. Unlike KL^XMA1B, the message is put in the "WASTE" basket.



REF: Compare this API to the KLQ^XMA1B: API described in this chapter and the DELMSG^XMXAPI(): Delete Messages from a Basket API described in Chapter 4, "Message Actions," in this manual.

Format KLQ^XMA1B

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables XMDUZ: (required) DUZ of the user who owns the basket.

XMK: (optional) Basket IEN. If '\$G(XMK)', then MailMan look to see in which basket the message is located.

XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Variables None

8.1.4.3 S2^XMA1B: Put a Message in a Basket

Reference Type Supported

Category Cross-category Activities—Mailboxes, Baskets, and Messages (Classic MailMan)

IA # 10065

Description This API puts a message in a basket; however, be careful with this call, because it does not check to see if the message is already in another basket—it just puts it where you tell it, so you could end up with the same message in more than one basket.



REF: Compare this API to the MOVEMSG^XMXAPI(): Move Messages to Another Basket API described in Chapter 4, "Message Actions," in this manual.

Format S2^XMA1B

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables XMDUZ: (required) DUZ of the user who owns the basket.
XMKM: (required) Basket IEN.
XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output Variables None

Variables KILLED Upon Exit XMKM

9.0 Mail Group Actions

9.1 ^XMA21

9.1.1 CHK^XMA21: Verify User's Mail Group Membership

Reference Type	Supported
Category	Mail Group Actions (Classic MailMan)
IA #	10067
Description	This API checks to see if a user is a member of a mail group. If the user is a member, \$T is set to "true"; otherwise, it's set to "false".
Format	CHK^XMA21

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (required) DUZ of the user in question. Y: (required) IEN of the mail group in the MAIL GROUP file (#3.8).
Output Variables	\$T: Results: True—User is a member of the mail group. False—User is <i>not</i> a member of the mail group.

9.2 ^XMBGRP

9.2.1 \$\$DM^XMBGRP(): Delete Local Members from a Mail Group

Reference Type	Supported
Category	Mail Group Actions (Classic MailMan)
IA #	1146
Description	This extrinsic function deletes local members from a mail group. Only local members can be deleted with this API. There is no API that deletes any other type of member.
Format	\$\$DM^XMBGRP(xmgroup,.xmy,xmquiet)
Input Parameters	xmgroup: (required) Mail group Internal Entry Number (IEN) or full name (without the G.).

.xmy: (required) Array of local users to delete from the mail group. Only DUZs are accepted (names are not).

xmquiet: (required) Silent flag. 1 (default)—Silent. Any errors will be sent in a message to the Postmaster and the user (DUZ) making this call. 0—Interactive. Any errors will be displayed.

Output returns: Returns:
Successful—1
Unsuccessful (error)—0

Parameters Killed Upon Exit xmy

9.2.2 \$\$MG^XMBGRP(): Create New Mail Group or Add Members to an Existing Mail Group

Reference Type Supported

Category Mail Group Actions (Classic MailMan)

IA # 1146

Description This extrinsic function creates a new mail group or adds members to an existing mail group. Only local members can be added with this API. There is no API that adds any other type of member.

Format \$\$MG^XMBGRP(xmgroup,xmtype,xmorg,xmself,.xmy,.xmdesc,xmquiet)

Input Parameters xmgroup: (required) This parameter can be either of the following: New Mail Group—Mail group name (*without* the G.). Existing Mail Group—Mail group Internal Entry Number (IEN) or full name (without the G.).
xmtype: (required) Type of mail group. Used only for creating a mail group, otherwise it's ignored. 0 (default)—Public 1—Private
xmorg: (required) DUZ of group organizer. Used only for creating a mail group, otherwise it's ignored.
xmself: (required) Allow self-enrollment?
0—No
1—Yes
Used only for creating a new mail group, otherwise it's ignored.
.xmy: (required) Array of local users to add to the mail group. Only DUZs are accepted (names are not).
.xmdesc: (required) Array of text to put in the description field of the mail group. Used only for creating a mail group, otherwise it's ignored.
xmquiet: (required) Silent flag. 1 (default)—Silent. Any errors will be sent in a message to the Postmaster and the user (DUZ) making this call. 0—Interactive. Any errors will be displayed.

Output returns: Returns:
Successful—Mail Group Internal Entry Number (IEN)
Unsuccessful (error)—0

Parameters Killed Upon Exit xmy

9.3 ^XMXAPIG

9.3.1 ADDMBRS^XMXAPIG(): Add Member(s) to Mail Group(s)

Reference Type	Supported
Category	Mail Group Actions
IA #	3006
Description	This API adds member(s) to mail group(s). Local users, devices, server options, mail groups, and remote users can be added to mail groups using this API; however, distribution lists, fax recipients, and fax groups are <i>not</i> handled by this API. Optionally, find and forward existing mail group messages to the local users.
Format	ADDMBRS^XMXAPIG(xmduz,xmgrp,xmmbrr[,xminstr][,xmtsk])
Input Parameters	xmduz: (required) The user DUZ who is adding the members to the mail group(s). The user must be authorized to edit the mail groups. Group coordinators or organizers may edit their own mail groups. The following users may edit public mail groups or unrestricted private groups: Clinical Application Coordinators Anyone possessing the XMMGR security key Anyone possessing the XM GROUP EDIT MASTER security key. xmgrp: (required) This is the mail group to be edited. It can be passed by value for a single group or by reference for one or more groups. It can be the IEN(s) or name of the mail group(s) in the MAIL GROUP file (#3.8). For example: XMGROUP="GROUP A" or XMGROUP("GROUP A")="" XMGROUP("GROUPB")="" xmmbrr: (required) The new member(s) to be added. It can be passed by value for one member or by reference for one or more members. The same rules apply for specifying xmmbrrs as apply when you are addressing a message. xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "FLAGS", "FDATE", "TDATE"
Output Parameter	xmtsk. The number of the TaskMan task that will find and forward past mail group messages to local users. It is returned only if XMINSTR("FLAGS") is passed in.

Example

```
>D ADDMBRS^XMXAPIG(XMDUZ,[.]XMGROUP,[.]XMMBR,.XMINSTR,.XMTSK)
```

9.3.2 DROP^XMXAPIG(): Drop Member from a Mail Group

Reference Type	Supported
Category	Mail Group Actions
IA #	3006
Description	This API is used when a user chooses to drop out of a mail group.
Format	DROP^XMXAPIG(xmduz,xmgroup)
Input Parameters	xmduz: (required) The user DUZ who wants to drop out of a mail group.

xmgroup: (required) This is the mail group from which the user wants to drop out. It can be the IEN or name of the mail group in the MAIL GROUP file (#3.8). For example:
 XMGROUP="GROUP A"

Output None

Example

```
>D DROP^XMXAPIG(XMDUZ,XMGROUP)
```

9.3.3 \$\$GOTLOCAL^XMXAPIG(): Check if a Mail Group has *Active* Local Members

Reference Type	Supported
Category	Mail Group Actions
IA #	3006
Description	This extrinsic function is used to find out whether or not a mail group has any <i>active</i> local members. Messages can only be delivered to <i>active</i> local members of a mail group. Active members are described as having an Access code and a mailbox. It sets XMERR and ^TMP("XMERR",\$J), if an error occurs.
Format	\$\$GOTLOCAL^XMXAPIG(xmgroup,xmdays)
Input Parameters	xmgroup: (required) Mail group Internal Entry Number (IEN) or name (exact, case-sensitive). xmdays: (optional) Active members of the mail group <i>must</i> have used MailMan within the past number of days specified by XMDAYS. If XMDAYS is 0, null, or not supplied, it is ignored.
Output	None Returns: 0—No <i>active</i> members in a local mail group. 1—Has <i>active</i> members in a local mail group.

Example 1

```
I '$$GOTLOCAL^XMXAPIG(XMGROUP,XMDAYS) D error
```

Example 2

```
I '$$GOTLOCAL^XMXAPIG("GROUP") D error
```

If the mail group named "GROUP" has no active local members, do an error routine to notify someone. Otherwise, go ahead and send the message.

Optionally, you can specify an additional constraint, that at least one member must have used MailMan in the last few days:

```
I '$$GOTLOCAL^XMXAPIG("GROUP",9) D error
```

If the mail group named "GROUP" does not have at least one active local member who has used MailMan in the last 9 days, do an error routine to notify someone. Otherwise, go ahead and send the message.

9.3.4 JOIN^XMXAPIG(): Enable User to Enroll in (Join) a Mail Group

Reference Type	Supported
Category	Mail Group Actions
IA #	3006
Description	This API enables a user to enroll in (join) a mail group. Optionally, find and forward existing mail group messages to the newly joined local user.
Format	JOIN^XMXAPIG(xmduz,xmgroup[,xminstr][,xmtsk])
Input Parameters	xmduz: (required) The user DUZ who wants to join the mail group. xmgroup: (required) This is the mail group the user wants to join. It can be the IEN or name of the mail group in the MAIL GROUP file (#3.8). For example: XMGROUP="GROUP A" xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "FLAGS", "FDATE", "TDATE"
Output Parameters	xmtsk. The number of the TaskMan task that will find and forward past mail group messages to local users. It is returned only if XMINSTR("FLAGS") is passed in.

Example

```
>D JOIN^XMXAPIG(XMDUZ,[.]XMGROUP,.XMINSTR,.XMTSK)
```


10.0 Bulletins—Creating and Sending

10.1 ^XMB

:

10.1.1 ^XMB: Create & Send a Bulletin in the Background

Reference Type	Supported
Category	Bulletins—Creating and Sending (Classic MailMan)
IA #	10069
Description	This API creates and sends a bulletin in the background. Unlike EN^XMB and SENDBULL^XMXAPI, the message number (XMZ) is not returned. The bulletin is sent to the mail groups defined for the bulletin in the BULLETIN file (#3.6), as well as to any additional recipients defined in XMY.



REF: Compare this API to the EN^XMB: API described in this chapter and the SENDBULL^XMXAPI(): Send a Bulletin and TASKBULL^XMXAPI(): Send a Bulletin APIs described in Chapter 4, "Message Actions," in this manual.

Format ^XMB

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables	<p>DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.</p> <p>XMTEXT: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. This is text, in addition to the text defined in the bulletin, to append to the bulletin.</p> <p>XMY: (optional) Recipients, in addition to those defined in the bulletin. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. If the bulletin has no recipients and there are no recipients in XMY, the bulletin will not be sent, and there will not be any error indication.</p>
Input Variables	XMB: (required) Full, exact name, of the bulletin. Case is important.

XMB(#): (optional) Bulletin parameter(s). For example:
 XMB(1)=<parm 1>, XMB(2)=<parm 2>, etc.

XMBTMP (optional) If \$D(XMBTMP) do not initialize (KILL) the ^TMP addressee global, because it contains bulletin addressees.

X MDF (optional) If \$D(X MDF) all addressing restrictions are waived.

XMDT (optional) Date/time (in any format understood by VA FileMan) to send the bulletin. The default is now.

XMYBLOB (optional) Specifically for the Imaging software.

Output Variables XMB: Results: Unchanged—If bulletin is found -1—If bulletin is not found in BULLETIN file (#3.6)

Variables KILLED Upon Exit XMTEXT, XMY

10.1.2 BULL^XMB: Create & Send a Bulletin (Interactive)

Reference Type	Supported
Category	Bulletins—Creating and Sending (Classic MailMan)
IA #	10069
Description	This API creates and sends a bulletin interactively. This is the same entry point called by the XMPOST option. This is a good way to test a bulletin.
Format	BULL^XMB

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.
 XMDUZ: (required) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

Output Variables None

10.1.3 EN^XMB: Create & Send a Bulletin in the Foreground

Reference Type	Supported
Category	Bulletins—Creating and Sending (Classic MailMan)
IA #	10069
Description	This API creates and sends a bulletin in the foreground. Unlike the ^XMB and TASKBULL^XMXAPI APIs, the message number (XMZ) is returned. The bulletin is sent to the mail groups defined for the bulletin in the BULLETIN file (#3.6), as well as to any additional recipients defined in XMY.



REF: Compare this API to the ^XMB API in this chapter and the SENDBULL^XMXAPI(): Send a Bulletin and TASKBULL^XMXAPI(): Send a Bulletin APIs described in Chapter 4, "Message Actions," in this manual.

Format EN^XMB

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Core Input Variables

DUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

XMDUZ: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

XMTEXT: (optional) For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. This is text, in addition to the text defined in the bulletin, to append to the bulletin.

XMY: (optional) Recipients, in addition to those defined in the bulletin. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

If the bulletin has no recipients and there are no recipients in XMY, the bulletin will not be sent, and there will not be any error indication.

Input Variables

XMB: (required) Full, exact name, of the bulletin. Case is important.

XMB(#): (optional) Bulletin parameter(s). For example:

XMB(1)=<parm 1>, XMB(2)=<parm 2>, etc.

XMBTMP: (optional) If \$D(XMBTMP) do not initialize (KILL) the ^TMP addressee global, because it contains bulletin addressees.

X MDF: (optional) If \$D(X MDF) all addressing restrictions are waived.

XMYBLOB: (optional) Specifically for Imaging software.

Output Variables

XMB: Results: Undefined—Bulletin is found
-1—Bulletin is *not* found in BULLETIN file (#3.6)

XMZ: Results:

Successful—Message number in the MESSAGE file (#3.9)

Unsuccessful—Unchanged or -1

Variables KILLED Upon Exit

XMB, XMTEXT, XMY

11.0 Address Lookup

11.1 ^XMA21

:

11.1.1 DES^XMA21: Address Lookup (Interactive, Next Default Recipient List)

Reference Type	Supported
Category	Address Lookup (Classic MailMan)
IA #	10067
Description	This API is an interactive address lookup, with the next default recipient set. Unlike DEST^XMA21, XMY is <i>not</i> KILLED upon entry.



REF: Compare this API to the DEST^XMA21: API described in this chapter and the TOWHOM^XMXAPIU(): API described in Chapter 13, "User Actions—Interactive," in this manual.

Format DES^XMA21

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (required) DUZ of the user doing the addressing. XMDF: (optional) If \$D(XMDF) all addressing restrictions are waived. XMMG: (optional) The DUZ or name of the next default recipient.
Output Variables	X: Results: "^"—If the user aborted addressing "" (Null)—If finished addressing normally XMOUT: If \$D(XMOUT), then the user aborted addressing. XMY: (required) Array of addressees. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

11.1.2 DEST^XMA21: Address Lookup (Interactive, First Default Recipient List)

Reference Type	Supported
Category	Address Lookup (Classic MailMan)
IA #	10067
Description	This API is an interactive address lookup, with the first default recipient set. Unlike DES^XMA21, XMY is KILLED upon entry.



REF: Compare this API to the DES^XMA21: API described in this chapter and the TOWHOM^XMXAPIU(): API described in Chapter 13, "User Actions—Interactive," in this manual.

Format DEST^XMA21

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (required) DUZ of the user doing the addressing. X MDF: (optional) If \$D(X MDF) all addressing restrictions are waived. X MDUN: (required) The name of the first default recipient.
Output Variables	X: Results: " ^"—If the user aborted addressing "" (Null)—If finished addressing normally X MOUT: If \$D(X MOUT), then the user aborted addressing. X MY: Array of addressees. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

11.1.3 INST^XMA21: Address Lookup (Non-Interactive)

Reference Type	Supported
Category	Address Lookup (Classic MailMan)
IA #	10067
Description	This API is a non-interactive address lookup. XMY is <i>not</i> KILLED upon entry.



REF: Compare this API to the DES^XMA21: API described in this chapter and the TOWHOM^XMXAPIU(): API described in Chapter 13, "User Actions—Interactive," in this manual.

Format INST^XMA21

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	X: (required) A local or remote address. (-X will remove any address.) X MDF: (optional) If \$D(X MDF) all addressing restrictions are waived. X MDUZ: (required) DUZ of the user doing the addressing. X MLOC: (optional) If \$D(X MLOC) then any error (in XMMG) will be displayed.
Output Variables	XMMG: Results: Error message—If Y=-1 "via <domain name>"—If remote address "" (Null)—Otherwise XMY: Array containing the address. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual. Y: Results of the address lookup: <DUZ^full name>—If local address <domain ien^domain name>—If remote address -1—If the lookup failed (bad address in X)

11.1.4 WHO^XMA21: Address Lookup (Non-Interactive)

Reference Type	Supported
Category	Address Lookup (Classic MailMan)
IA #	10067
Description	This API is a non-interactive address lookup. XMY is <i>not</i> KILLED upon entry.



REF: Compare this API to the DES^XMA21: API described in this chapter and the TOWHOM^XMXAPIU(): API described in Chapter 13, "User Actions—Interactive," in this manual.

Format	WHO^XMA21
---------------	-----------

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Address Lookup

Input Variables

X: (required) A local or remote address. (-X will remove any address.)

X MDF: (optional) If \$D(X MDF) all addressing restrictions are waived.

X MDUZ: (required) DUZ of the user doing the addressing.

X MLOC: (optional) If \$D(X MLOC) then any error (in X MMG) will be displayed.

Output Variables

X MMG: Results:

Error message—If Y=-1

"via <domain name>"—If remote address

"" (Null)—Otherwise

X MY: Array containing the address. For a description of this variable, please refer to Table 1-1 in the "Common Variables" topic in Chapter 1, "Introduction," in this manual.

Y: Results of the address lookup:

<DUZ^full name>—If local address

<domain ien^domain name>—If remote address

-1—If the lookup failed (bad address in X)

12.0 User Information

The following utility sets up the user's XMV array, with vital user information, user preferences, and, if the user is a surrogate, determining level of authorization:

12.1 ^XMVVITAE

12.1.1 INIT^XMVVITAE(): Set Up Vital User Information

Reference Type	Supported
Category	User Information
IA #	2728
Description	This API sets up vital user information and should <i>not</i> be used for any other purpose.



NOTE: This API is meant to be called once upon entry into MailMan.

Format	INIT^XMVVITAE
Input Parameters	DUZ: (required) User DUZ. XMDUZ: (optional) User (default) or surrogate DUZ.

The following are the Output Parameters

Output Parameters	Description	Meaning
XMDUZ:	Set to DUZ if XMDUZ was not supplied as an input parameter.	
	XMV:	Array of values:
	("ASK BSKT")	Ask basket if send message to self? <ul style="list-style-type: none">• 0—No• 1—Yes
	("BANNER")	User's banner
	("DUZ NAME")	User's real name
	("ERROR",1)	"You do not have a DUZ."
	("ERROR",2)	"There is no person with DUZ "_XMDUZ_"."
	("ERROR",3)	"There is no Access Code for DUZ "_XMDUZ_"."
	("ERROR",4)	"There is no mailbox for DUZ "_XMDUZ_"."
	("LAST USE")	Date/time user last entered MailMan (dd mmm yy hh:mm)
	("MSG DEF")	User's default message action: <ul style="list-style-type: none">• I—Ignore• D—Delete

Output Parameters	Description	Meaning
	("NAME")	User's real name, or name of surrogate
	("NETNAME")	User's name as it will appear on messages sent to remote sites
	("NEW MSGS")	Number of new messages
	("NOSEND")	User can or cannot send messages (multiple logon): <ul style="list-style-type: none"> • 0—Can send messages • 1—Cannot send messages
	("ORDER")	User's preferred message display order: <ul style="list-style-type: none"> • 1—Chronological • -1—Reverse chronological
	("PRIV")	Contains surrogate privileges: <ul style="list-style-type: none"> • "R"—Read • "W"—Write
	("RDR ASK")	Ask user which message reader to use? <ul style="list-style-type: none"> • 0—No • 1—Yes
	("RDR DEF")	User's default message reader: <ul style="list-style-type: none"> • C (Classic) • D—Detailed full screen • S—Summary full screen
	("SHOW INST")	Show user's institution? <ul style="list-style-type: none"> • 0—No • 1—Yes
	("SHOW TITL")	Show user's title? <ul style="list-style-type: none"> • 0—No • 1—Yes
	("SYSERR",i)	<text> Domain incorrectly set up.
	("VERSION")	"VA MailMan "_<version number>
	("WARNING",1)	"Priority Mail"
	("WARNING",2)	"Message in Buffer"
	("WARNING",3)	"No Introduction"
	("WARNING",4)	"Multiple Signon"
	("WARNING",5)	

Output Parameters	Description	Meaning
XMDISPI:	"^" Piece 1 contains any of following: <ul style="list-style-type: none"> "T"—Show titles "A"—Ask basket "I"—Show institution "^" Piece 2 contains either of following message action defaults: <ul style="list-style-type: none"> "I"—Ignore "D"—Delete 	
XMDUN:	Same as XMV("NAME").	
XMNOSEND:	Equals 1 if XMV("NOSEND")=1, otherwise not defined.	
XMPRIV:	Surrogate READ (y/n) "^" WRITE (send, y/n) privilege.	Example: "y^n"—User can read, but <i>not</i> send messages

12.1.2 OTHER^XMOVITAE: Change User Settings When User Becomes a Surrogate

Reference Type	Supported
Category	User Information
IA #	2728
Description	This API changes certain MailMan settings when the user becomes a surrogate.



NOTE: This API is meant to be called whenever the user becomes a surrogate.

Format OTHER^XMOVITAE

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables
 DUZ: (required) User DUZ.
 XMDUZ: (required) Surrogate DUZ.

The following are the Output Parameters

Output Parameters	Description	Meaning
XMV:	Array of values, all pertaining to the user signified by XMDUZ. Any XMV variables not listed here are unchanged.	
	("LAST USE")	Date/time user last entered MailMan: dd mmm yy hh:mm
	("NAME")	Name of user.
	("NETNAME")	User's name as it will appear on messages sent to remote sites.
	("NEW MSGS")	Number of new messages.
	("NOSEND")	User can (=0) or <i>cannot</i> (=1) send messages (multiple logon).
	("PRIV")	Contains surrogate privileges ("R" READ, "W" WRITE).
	The following XMV variables are KILLed and then set only if conditions warrant:	
	("BANNER")	User's banner.
	("ERROR",1)	"You do not have a DUZ."
	("ERROR",2)	"There is no person with DUZ "_XMDUZ_"."
	("ERROR",3)	"There is no Access Code for DUZ "_XMDUZ_"."
	("ERROR",4)	"There is no Mail Box for DUZ "_XMDUZ_"."
	("WARNING",1)	"Priority Mail".
	("WARNING",2)	"Message in Buffer".
	("WARNING",3)	"No Introduction".
	("WARNING",4)	"Multiple Signon".
	("WARNING",5)	"POSTMASTER has "_I_" baskets." (if more than 900).
XMDUN:	Same as XMV("NAME").	
XMNOSEND:	Equals 1 if XMV("NOSEND")=1, otherwise not defined.	
XMPRIV:	Surrogate READ (y/n) "^" WRITE (send, y/n) privilege.	Example: "y^n" means that the user can read, but not send messages.

12.1.3 SELF^XMMVITAE: Restore Certain MailMan Settings Once User No Longer a Surrogate

Reference Type Supported
Category User Information
IA # 2728

Description This API restores certain MailMan settings when the user becomes himself/herself again, after having been a surrogate.



NOTE: This API is meant to be called whenever the user returns from being a surrogate.

Format SELF^XMMVITAE

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables DUZ: (required) User DUZ.

Variables KILLed Upon Exit XMV("PRIV"),
XMPRIV

The following are the Output Parameters

Output Parameters	Description	Meaning
XMDUZ:	User DUZ.	
XMV:	Array of values, all pertaining to the user signified by DUZ. Any XMV variables not listed here are unchanged.	
	("NAME")	Name of user.
	("NETNAME")	User's name as it will appear on messages sent to remote sites.
	("NEW MSGS")	Number of new messages.
	("NOSEND")	User can (=0) or <i>cannot</i> (=1) send messages (multiple logon)
	The following XMV variables are KILLed and then set only if conditions warrant:	
	("BANNER")	User's banner.
	("ERROR",1)	"You do not have a DUZ."
	("ERROR",2)	"There is no person with DUZ "_XMDUZ_"."
	("ERROR",3)	"There is no Access Code for DUZ "_XMDUZ_"."
	("ERROR",4)	"There is no Mailbox for DUZ "_XMDUZ_"."
	("WARNING",1)	"Priority Mail".
	("WARNING",2)	"Message in Buffer".

Output Parameters	Description	Meaning
	("WARNING",3)	"No Introduction".
	("WARNING",4)	"Multiple Signon".
	("WARNING",5)	"POSTMASTER has "_I_" baskets." (if more than 900).
XMDUN:	Same as XMV("NAME").	
XMNOSEND:	Equals 1 if XMV("NOSEND")=1, otherwise not defined.	

13.0 User Actions—Interactive

13.1 ^XM

These APIs can be used to create menu options.

The primary option should be set up as follows:

```
Entry action:  S XMMENU(0)=<name of the menu option> D EN^XM
Routine:      xxx^XMXAPIU
Exit action:   K XMMENU D CHECKOUT^XM
```

Any subordinate option should be set up as follows:

```
Entry action:  D CHECKIN^XM
Routine:      xxx^XMXAPIU
Exit action:   D CHECKOUT^XM
```

13.1.1 CHECKIN^XM: Entry Action for Any Subordinate MailMan Option

Reference Type	Supported
Category	User Actions—Interactive
IA #	10064
Description	This API is meant to be the entry action of any subordinate MailMan option.
Format	CHECKIN^XM
Input Parameters	None
Output	None

13.1.2 CHECKOUT^XM: Exit Action for Any Subordinate MailMan Option

Reference Type	Supported
Category	User Actions—Interactive
IA #	10064
Description	This API is meant to be the exit action of every MailMan option.
Format	CHECKOUT^XM
Input Parameters	None
Output	None

13.1.3 EN^XM: Entry Action of the Primary MailMan Option—Set Up Environment

Reference Type	Supported
Category	User Actions—Interactive (Classic MailMan)
IA #	10064

Description	This API is meant to be the entry action of the primary MailMan option. It sets up the user's MailMan environment and calls the HEADER^XM: API to greet the user.
Format	EN^XM
Input Parameters	None
Output	None

13.1.4 HEADER^XM: Entry Action of the Primary MailMan Option—Display User Greeting

Reference Type	Supported
Category	User Actions—Interactive
IA #	10064
Description	This API is meant to be part of the entry action of the primary MailMan option, whether called by itself, or as part of another call, such as EN^XM: Entry Action of the Primary MailMan Option. It displays a greeting to the user.
Format	HEADER^XM
Input Parameters	None
Output	None

13.2 ^XMXAPIU

The following are meant to be in an option's ROUTINE field. They expect that DUZ exists, and if the user is acting as a surrogate, that XMDUZ exists too. Otherwise, XMDUZ will be set to DUZ. If the XMV variables do not exist, INIT^XMVITAE will be called.

13.2.1 READ^XMXAPIU: Read/Manage messages in a mailbox

Reference Type	Supported
Category	User Actions—Interactive
IA #	2774
Description	This API reads/manages messages in a mailbox.



NOTE: Only the user or a surrogate can use this API.



REF: See also: REC^XMA: Read/Manage Messages (Interactive) in Chapter 8, "Cross-category Activities—Mailboxes, Baskets, and Messages," in this manual.

Format READ^XMXAPIU

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (optional) The user whose mailbox is to be read. The default is DUZ. XMV: (optional) The user's variables.
Output Variables	None

13.2.2 READNEW^XMXAPIU: Read New Messages in a Mailbox

Reference Type	Supported
Category	User Actions—Interactive
IA #	2774
Description	This API reads new messages in a mailbox.



NOTE: Only the user or a surrogate can use this API.

Format	READNEW^XMXAPIU
---------------	-----------------

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (optional) The user whose new messages are to be read. The default is DUZ. XMV: (optional) The user's variables.
Output Variables	None

13.2.3 SEND^XMXAPIU: Send a Message

Reference Type	Supported
Category	User Actions—Interactive
IA #	2774
Description	This API sends a message.



NOTE: Only the user or a surrogate with "WRITE" privileges can use this API.

Format	SEND^XMXAPIU
---------------	--------------

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (optional) The user who is to send a message. The default is DUZ. XMV: (optional) The user's variables.
Output Variables	None

13.2.4 TOWHOM^XMXAPIU(): Ask User for Message Addressees

Reference Type	Supported
Category	User Actions—Interactive
IA #	2774
Description	This API asks a user for message addressees.



NOTE: This API is meant to be used in a routine.

Format	TOWHOM^XMXAPIU(xmduz,xmz,xmtype[,..xminstr])
Input Parameters	xmduz: (required) The user (DUZ or name) who is addressing the message. xmz: (required) Message number in the MESSAGE file (#3.9). This is not necessary if XMCTYPE="S" and XMINSTR("ADDR FLAGS") contains "R". xmtype: (required) Determines what prompts are used with the user: S—User is sending a message. F—User is forwarding a message. .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "ADDR FLAGS", "TO PROMPT"
Output Parameters	.xminstr: Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "SELF BSKT", "SHARE BSKT", "SHARE DATE"

The following global variables are created:

- ^TMP("XMY0", \$J) Addressee as entered by the user.
- ^TMP("XMY", \$J) Resulting addressee(s) as interpreted by MailMan.

14.0 Security—Permissions and Restrictions

14.1 Errors

If any errors occur, the following variables will be defined:

XMERR The number of errors.

`^TMP("XMERR", $J, <error number>, "TEXT", <line number>) = <error text>`

14.2 ^XMXSEC

14.2.1 \$\$ACCESS^XMXSEC(): Check if User Can Access a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can access a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR", \$J), if access/permission is denied.
Format	<code>\$\$ACCESS^XMXSEC(xmduz, xmz[, xmzrec])</code>
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9, XMZ, 0).
Output	returns: Returns: 0—No, the user <i>cannot</i> access a message. 1—Yes, the user <i>can</i> access a message.

14.2.2 \$\$ANSWER^XMXSEC(): Check if User Can Answer a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can answer a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR", \$J), if access/permission is denied.
Format	<code>\$\$ANSWER^XMXSEC(xmduz, xmz[, xmzrec])</code>
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9, XMZ, 0).
Output	returns: Returns: 0—No, the user <i>cannot</i> answer a message. 1—Yes, the user <i>can</i> answer a message.

14.2.3 \$\$BCAST^XMXSEC(): Check if Message was Broadcast

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message was broadcast or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).
Format	\$\$BCAST^XMXSEC(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message was <i>not</i> broadcast. 1—Yes, the message was broadcast.

14.2.4 \$\$CLOSED^XMXSEC(): Check if Message is "Closed"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Closed" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



NOTE: Compare this API to the \$\$ZCLOSED^XMXSEC(): API described in this chapter.

Format	\$\$CLOSED^XMXSEC(xmzrec)
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, the message is <i>not</i> "Closed." 1—Yes, the message is "Closed."

14.2.5 \$\$CONFID^XMXSEC(): Check if Message is "Confidential"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Confidential" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$ZCONFID^XMXSEC(): API described in this chapter.

Format	\$\$CONFID^XMXSEC(xmzrec)
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, the message is <i>not</i> "Confidential." 1—Yes, the message is "Confidential."

14.2.6 \$\$CONFIRM^XMSEC(): Check if Message is "Confirm Receipt Requested"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Confirm Receipt Requested" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$ZCONFID^XMSEC(): API described in this chapter.

Format	\$\$CONFIRM^XMSEC(xmzrec)
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, the message is <i>not</i> "Confirm Receipt Requested." 1—Yes, the message is "Confirm Receipt Requested."

14.2.7 \$\$COPY^XMSEC(): Check if User Can Copy a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can copy a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$COPY^XMSEC(xmduz,xmz[,xmzrec])
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, user <i>cannot</i> copy a message. 1—Yes, user can copy a message.

14.2.8 \$\$DELETE^XMSEC(): Check if User Can Delete/Terminate a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can delete or terminate a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$DELETE^XMSEC(xmduz,xmk,xmz[,xmzrec])
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN.

xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).

Output

returns: Returns:

0—No, user *cannot* delete/terminate a message.

1—Yes, user can delete/terminate a message.

14.2.9 \$\$FORWARD^XMXSEC(): Check if User Can Forward a Message

Reference Type Supported

Category Security—Permissions and Restrictions

IA # 2731

Description This extrinsic function returns a value indicating whether the user can forward a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.

Format \$\$FORWARD^XMXSEC(xmduz,xmz,xmzrec)

Input Parameters xmduz: (required) User DUZ.

xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).

Output returns: Returns:

0—No, user *cannot* forward a message.

1—Yes, user can forward a message.

14.2.10 \$\$INFO^XMXSEC(): Check if Message is "Information Only"

Reference Type Supported

Category Security—Permissions and Restrictions

IA # 2731

Description This extrinsic function returns a value indicating whether a message is "Information Only" or not (0 = No; 1 = Yes). It does *not* set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$ZINFO^XMXSEC(): API described in this chapter.

Format \$\$INFO^XMXSEC(xmzrec)

Input Parameters xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).

Output returns:

Returns:

0—No, the message is not "Information Only."

1—Yes, the message is "Information Only."

14.2.11 \$\$LATER^XMXSEC(): Check if User Can "Later" a Message

Reference Type Supported

Category Security—Permissions and Restrictions

IA # 2731

Description This extrinsic function returns a value indicating whether the user can "later" a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.

Format \$\$LATER^XMXSEC(xmduz)

Input Parameters xmduz: (required) User DUZ.
Output returns:
 Returns:
 0—No, user *cannot* "later" a message.
 1—Yes, user can "later" a message.

14.2.12 \$\$MOVE^XMXSEC(): Check if User Can Save or Filter a Message

Reference Type Supported
Category Security—Permissions and Restrictions
IA # 2731
Description This extrinsic function returns a value indicating whether the user can save or filter a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format \$\$MOVE^XMXSEC(xmduz,xmz[,xmzrec])
Input Parameters xmduz: (required) User DUZ.
 xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output returns:
 Returns:
 0—No, user *cannot* save or filter a message.
 1—Yes, user can save or filter a message.

14.2.13 \$\$ORIGIN8R^XMXSEC(): Check if User Sent a Message

Reference Type Supported
Category Security—Permissions and Restrictions
IA # 2731
Description This extrinsic function returns a value indicating whether the user (XMDUZ or DUZ) sent the message or not (sender or surrogate, 0 = No; 1 = Yes). It does *not* set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$ZORIGIN8^XMXSEC(): API described in this chapter

Format \$\$ORIGIN8R^XMXSEC(xmduz,xmzrec)
Input Parameters xmduz: (required) User DUZ.
 xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output returns:
 Returns:
 0—No, user did *not* send the message.
 1—Yes, user did send the message.

14.2.14 \$\$POSTPRIV^XMXSEC: Check if User has Postmaster Privileges

Reference Type Supported
Category Security—Permissions and Restrictions

IA #	2731
Description	This extrinsic function returns a value indicating whether the user has Postmaster privileges or not, including whether or not the user can perform group message actions in SHARED,MAIL (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$POSTPRIV^XMXSEC
Input Parameters	None
Output	returns: Returns 0—No, user does <i>not</i> have Postmaster privileges. 1—Yes, user does have Postmaster privileges.

14.2.15 \$\$PRIORITY^XMXSEC(): Check if Message is "Priority"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Priority" or not (0 = No; 1 = Yes). It does not set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$ZPRI^XMXSEC(): API described in this chapter.

Format	\$\$PRIORITY^XMXSEC(xmzrec)
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, the message is <i>not</i> "Priority." 1—Yes, the message is "Priority."

14.2.16 \$\$READ^XMXSEC(): Check if User Can Read a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can read a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$READ^XMXSEC(xmduz,xmz[,xmzrec])
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, user <i>cannot</i> read a message. 1—Yes, user can read a message.

14.2.17 \$\$REPLY^XMXSEC(): Check if User Can Reply to a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can reply to a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$REPLY^XMXSEC(xmduz,xmz[,xmzrec])
Input Parameters	xmduz: (required) User DUZ. xmz: (Required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, user <i>cannot</i> reply to a message. 1—Yes, user can reply to a message.

14.2.18 \$\$RPRIV^XMXSEC(): Check if Surrogate has READ Privileges

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the surrogate has READ privileges or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$RPRIV^XMXSEC
Input Parameters	None
Output	returns: Returns: 0—No, surrogate does <i>not</i> have READ privileges. 1—Yes, surrogate does have READ privileges.

14.3 \$\$RWPRIV^XMXSEC(): Check if Surrogate has READ or WRITE Privileges

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the surrogate has READ or WRITE privileges or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$RWPRIV^XMXSEC
Input Parameters	None
Output	returns: Returns: 0—No, surrogate does <i>not</i> have READ or WRITE privileges. 1—Yes, surrogate does have READ or WRITE privileges.

14.3.1 \$\$SEND^XMSEC(): Check if User Can Send a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user can send a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$SEND^XMSEC(xmduz[,xminstr])
Input Parameters	xmduz: (required) User DUZ. .xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "FROM"
Output	returns: Returns: 0—No, user <i>cannot</i> send a message. 1—Yes, user can send a message.

14.3.2 \$\$SURRACC^XMSEC(): Check if Surrogate Can Access a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the surrogate can access a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$SURRACC^XMSEC(xmduz,xmaccess,xmz,xmzrec)
Input Parameters	xmduz: (required) User DUZ. xmaccess: (required) String telling type of access attempted. (Used in an error message, if access is denied.) xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, surrogate <i>cannot</i> access a message. 1—Yes, surrogate can access a message.

14.3.3 \$\$SURRCONF^XMSEC(): Check if Message is "Confidential" & Surrogate Access

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Confidential" or not, and if it is, whether the surrogate can access it (0 = No; 1 = Yes, it is confidential and the surrogate <i>cannot</i> access it). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



NOTE: This function should only be used when the user is a surrogate.

Format	\$\$SURRCONF^XMXSEC(xmduz,xmz)
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message is <i>not</i> "Confidential," so a surrogate can access the message. 1—Yes, the message is "Confidential," so a surrogate <i>cannot</i> access the message.

14.3.4 \$\$WPRIV^XMXSEC: Check if Surrogate has WRITE Privileges

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the surrogate has WRITE privileges or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR", \$J), if access/permission is denied.
Format	\$\$WPRIV^XMXSEC
Input Parameters	None
Output	returns: Returns: 0—No, surrogate does <i>not</i> have WRITE privileges. 1—Yes, surrogate does have WRITE privileges.

14.3.5 \$\$ZCLOSED^XMXSEC(): Check if Message is "Closed"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Closed" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR", \$J).



REF: Compare this API to the \$\$CLOSED^XMXSEC(): API described in this chapter.

Format	\$\$ZCLOSED^XMXSEC(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message is not "Closed." 1—Yes, the message is "Closed."

14.3.6 \$\$ZCONFID^XMXSEC(): Check if Message is "Confidential"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Confidential" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$CLOSED^XMXSEC(): API described in this chapter.

Format	\$\$ZCONFID^XMXSEC(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message is <i>not</i> "Confidential." 1—Yes, the message is "Confidential."

14.3.7 \$\$ZCONFIRM^XMXSEC(): Check if Message is "Confirm Receipt Requested"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Confidential" or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$CLOSED^XMXSEC(): API described in this chapter.

Format	\$\$ZCONFID^XMXSEC(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message is <i>not</i> "Confirm Receipt Requested." 1—Yes, the message is "Confirm Receipt Requested."

14.3.8 \$\$ZINFO^XMXSEC(): Check if Message is "Information Only"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Information Only" or not (0 = No; 1 = Yes). It does not set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the `$$INFO^XMXSEC()`: API described in this chapter.

Format	<code>\$\$ZINFO^XMXSEC(xmz)</code>
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, the message is <i>not</i> "Information Only." 1—Yes, the message is "Information Only."

14.3.9 `$$ZORIGIN8^XMXSEC()`: Check if User Sent a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user (XMDUZ or DUZ) sent the message or not (sender or surrogate, 0 = No; 1 = Yes). It does <i>not</i> set XMERR and <code>^TMP("XMERR",\$J)</code> .



REF: Compare this API to the `$$ORIGIN8R^XMXSEC()`: API described in this chapter.

Format	<code>\$\$ZORIGIN8^XMXSEC(xmduz,xmz)</code>
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0—No, user did <i>not</i> send the message. 1—Yes, user did send the message.

14.3.10 `$$ZPOSTPRV^XMXSEC`: Check if User has Postmaster Privileges

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether the user has Postmaster privileges or not, including whether or not the user can perform group message actions in SHARED,MAIL (0 = No; 1 = Yes). It does not set XMERR and <code>^TMP("XMERR",\$J)</code> .



REF: Compare this API to the `$$POSTPRIV^XMXSEC`: Check if User has Postmaster Privileges API described in this chapter.

Format	<code>\$\$ZPOSTPRV^XMXSEC</code>
Input Parameters	None
Output	returns: Returns: 0— No, user does not have Postmaster privileges." 1— Yes, user does have Postmaster privileges."

14.3.11 \$\$ZPRI^XMXSEC(): Check if Message is "Priority"

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2731
Description	This extrinsic function returns a value indicating whether a message is "Priority" or not (0 = No; 1 = Yes). It does not set XMERR and ^TMP("XMERR",\$J).



REF: Compare this API to the \$\$PRIORITY^XMXSEC(): API described in this chapter.

Format	\$\$ZPRI^XMXSEC(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 0— No, the message is not "Priority." 1— Yes, the message is "Priority."

14.4 ^XMXSEC1

14.4.1 CHKLINES^XMXSEC1(): Check if Message is Too Long to be Sent to a Remote Site

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This API checks whether a message is too long to be sent to a remote site. If \$D(XMRESTR("NONET")), then it is. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



Note: This routine does *not* KILL XMRESTR.

Format	CHKLINES^XMXSEC1(xmduz,xmz)
---------------	-----------------------------

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Parameters	xmduz: (required) User DUZ.
-------------------------	-----------------------------

Output Variables xnz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 XMRESTR("NONET") If the message is not too long or if the user holds the
 XMMGR security key, then XMRESTR("NONET") is *not* set. Otherwise,
 XMRESTR("NONET") equals the maximum number of lines a message can have
 when sending to a remote site.

14.4.2 CHKMSG^XMXSEC1(): Check Message Location

Reference Type Supported
Category Security—Permissions and Restrictions
IA # 2732
Description This API checks whether or not the message is located where the calling routine
 says it is, and whether or not the user can access it. It sets XMERR and
 ^TMP("XMERR",\$J), if access/permission is denied.
Format CHKMSG^XMXSEC1(xmduz,xmk,xmkz,xnz,xnzrec)
Input Parameters xmduz: (required) DUZ of the user who is accessing the message.
 xmk: (required) Message being accessed. For a description of this parameter,
 please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions,"
 in this manual.
 xmkz: (required) Message being accessed. For a description of this parameter,
 please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions,"
 in this manual.
Output Parameters xnz: Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xnzrec: Zero node of the message: ^XMB(3.9,XMZ,0).

14.4.3 \$\$COPYAMT^XMXSEC1(): Check Total Number of Lines & Responses to be Copied

Reference Type Supported
Category Security—Permissions and Restrictions
IA # 2732
Description This extrinsic function can be used when copying a message. It checks the total
 number of lines and responses to be copied. It returns 1 if the amount is within site
 limitations; 0, if not. It sets XMERR and ^TMP("XMERR",\$J), if permission is denied.
Format \$\$COPYAMT^XMXSEC1(xnz[,xmwhich])
Input Parameters xnz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xmwhich: (optional) String of which responses are to be copied. Options include:
 0—Original message
 Number List/Range—These particular responses
 Undefined/Null—Original message and all responses
Output returns:
 Returns:
 1—Amount is within site limitations.
 0—Amount is *not* within site limitations.

14.4.4 \$\$COPYLIMS^XMXSEC1: Get Message Copy Limits

Reference Type Supported

Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function can be used when copying a message. It returns the site's message copy limits (i.e., three-piece ^-delimited string): Piece 1: Number of recipients to whom the copy can be sent (default = 2999). Piece 2: Number of responses that can be copied (default = 99). Piece 3: Number of lines of text that can be copied (default = 3999). If the site has no specific limit, then MailMan defaults are used. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J)
Format	\$\$COPYLIMS^XMXSEC1
Input Parameters	None
Output	returns: Returns a three-piece ^-delimited string: Piece 1: Number of recipients to whom the copy can be sent (default = 2999). Piece 2: Number of responses that can be copied (default = 99). Piece 3: Number of lines of text that can be copied (default = 3999).

14.4.5 \$\$COPYRECP^XMXSEC1(): Check Total Number of Recipients on a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function can be used when copying a message. It checks the total number of recipients on the message to see if it's "OK" to list them in the copy text and send the copy to them, too. It returns 1 if the amount is within site limitations; 0, if not. It sets XMERR and ^TMP("XMERR",\$J), if permission is denied.
Format	\$\$COPYRECP^XMXSEC1(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: 1—Amount is within site limitations. 0—Amount is <i>not</i> within site limitations.

14.4.6 GETRESTR^XMXSEC1(): Get Sending/Forwarding Message Restrictions

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This API returns assorted restrictions, if any, on sending or forwarding the message. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).



Note: This routine does *not* KILL XMRESTR.

Format	GETRESTR^XMXSEC1(xmduz,xmz[,xmzrec][,xminstr],xmrestr)
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).

.xminstr: (optional) Appropriate special instructions. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "ADDR FLAGS"

Output Parameters

.xmrestr: Restrictions on forwarding the message. Here are the nodes that can be set:

If \$G(XMRESTR("NONET")), then the message is too long to be sent to a remote site, and it equals the maximum number of lines a message can have when sending to a remote site.

If \$G(XMRESTR("FLAGS"))["C"], then the message *cannot* be forwarded to SHARED,MAIL (because it is confidential).

If \$G(XMRESTR("FLAGS"))["X"], then the message *cannot* be forwarded to SHARED,MAIL (because it is closed).

If \$D(XMRESTR("NOFPG")), then the message *cannot* be forwarded to groups (because it is a priority message, the user did not send it, the user does not possess the XM GROUP PRIORITY security key, and XMINSTR("ADDR FLAGS")["R"]).

14.4.7 OPTGRP^XMXSEC1(): Determine User Capabilities at Basket/Message Group Level

Reference Type Supported
Category Security—Permissions and Restrictions
IA # 2732
Description This API determines what the user can do at the basket or message group level. It does *not* set XMERR and ^TMP("XMERR",\$J).
Format OPTGRP^XMXSEC1(xmduz,xmk,.xmopt)
Input Parameters xmduz: (required) DUZ of the user who is accessing the basket.
 xmk: (required) Basket IEN.

The Output is shown in the following table:

Output	Description	Meaning
.xmopt:	Commands that the user can use are a subset of the following:	
	If the user cannot use them, they either will not appear (only the Postmaster will see "X") or they will have subnode(s) under "?" with an explanation as to why they cannot be used.	
	For example, if the basket were the "IN" basket, then XMOPT("C","?")="The name of the IN basket cannot be changed."	
	If there were n lines of text, then XMOPT("C","?",1), ... XMOPT("C","?",n-1), and XMOPT("C","?") would be set.	
	("C")	"Change the name of this basket"
	("D")	"Delete messages"

Output	Description	Meaning
	("F")	"Forward messages"
	("FI")	"Filter messages"
	("H")	"Headerless Print messages"
	("L")	"Later messages"
	("N")	"New messages list"
	("P")	"Print messages"
	("Q")	"Query (search for) messages"
	("R")	"Resequence messages"
	("S")	"Save messages"
	("T")	"Terminate messages"
	("X")	"Xmit priority toggle"

14.4.8 \$\$PAKMAN^XMXSEC1(): Check if PackMan Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function returns a value indicating whether a message is a PackMan message or not (0 = No; 1 = Yes). It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).
Format	\$\$PAKMAN^XMXSEC1(xmz[,xmzrec])
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0—No, the message is <i>not</i> a PackMan message. 1—Yes, the message is a PackMan message.

14.4.9 \$\$\$SPRIV^XMXSEC1: Check if User Authorized to Conduct Super Search

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function returns a value indicating if a user is authorized to conduct a Super Search. If not, it also sets XMERR and ^TMP("XMERR",\$J).
Format	\$\$\$SPRIV^XMXSEC1
Input Parameters	None
Output	returns: Returns: 0— User cannot conduct a Super Search. 1— User can conduct a Super Search

14.4.10 \$\$ZSSPRIV^XMXSEC1: Check if User Authorized to Conduct Super Search

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function returns a value that indicates if a user is authorized to conduct a Super Search. If not, it does not set XMERR and ^TMP("XMERR",\$J).
Format	\$\$\$SSPRIV^XMXSEC1
Input Parameters	None
Output	returns: Returns: 0— User cannot conduct a Super Search. 1— User can conduct a Super Search

14.5 ^XMXSEC2

14.5.1 \$\$EDIT^XMXSEC2(): Check if User Can Edit a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2732
Description	This extrinsic function returns a value indicating whether the user can edit a message or not (0 = No; 1 = Yes). It sets XMERR and ^TMP("XMERR",\$J), if access/permission is denied.
Format	\$\$EDIT^XMXSEC2(xmduz,xmz,xmzrec)
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns: 0— No, user cannot edit a message. 1— Yes, user can edit a message.

14.5.2 OPTEDIT^XMXSEC2(): Determine What the User Edit

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2733
Description	If the OPTMSG^XMXSEC2(): this API determines that the user can edit the message, then the OPTEDIT^XMXSEC2 API determines what, exactly, the user can edit. It does <i>not</i> set XMERR and ^TMP("XMERR",\$J).
Format	OPTEDIT^XMXSEC2(.xminstr,.xmopt)
Input Parameters	.xminstr: (required) Set by INMSG2^XMXUTIL2: "FLAGS", "TYPE", "VAPOR", "RCPT BSKT", "SCR HINT"

Output Parameters Commands that the user can use are a subset of the following:
 If the user *cannot* use them, they will have subnode(s) under "?" with an explanation as to why they *cannot* be used.

- ("C") "Confidential set/remove"
- ("D") "Delivery basket set/remove"
- ("P") "Priority/Normal message"
- ("R") "Confirm receipt set/remove"
- ("S") "edit Subject"
- ("T") "edit Text"
- ("V") "Vaporize date set/remove"
- ("X") "Closed message set/remove"

14.5.3 OPTMSG^XMXSEC2(): Determine What the User Can do with a Message

Reference Type	Supported
Category	Security—Permissions and Restrictions
IA #	2733
Description	This API determines what the user can do with the message. Some input parameters are set by calling INMSG1 and INMSG2^XMXUTIL2. It does <i>not</i> set XMERR and ^TMP("XMERR", \$J).
Format	OPTMSG^XMXSEC2(xmduz,xmk,xmz,.xmim,.xminstr,.xmiu,.xmopt)
Input Parameters	<p>xmduz: (required) DUZ of the user who is accessing the message.</p> <p>xmk: (required) Basket IEN where the message is located: 0—If not in basket</p> <p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>.xmim: (required) Message information, set by INMSG1^XMXUTIL2: ("FROM") Who sent the message.</p> <p>.xminstr: (required) Set by INMSG2^XMXUTIL2 "FLAGS", "TYPE", "VAPOR", "RCPT BSKT", "SCR HINT"</p> <p>.xmiu: (required) User information, as related to the message: ("ORIGN8") Did the user send the message? Set by INMSG2^XMXUTIL2. ("IEN") User IEN in message RECIPIENT multiple, set by INMSG1^XMXUTIL2.</p>

Output Parameters

.xmopt: Commands that the user can use are a subset of the following:
If the user *cannot* use them, they will have subnode(s) under "?" with an explanation as to why they *cannot* be used.

For example, if the message is Information Only, then XMOPT("R","?")="Only the sender can Reply to an 'Information only' message."

If there were *n* lines of text, then XMOPT("R","?",1),

XMOPT("R","?",*n*-1), and XMOPT("R","?") would be set.

("A") "Answer"
 ("AA") "Access Attachments"
 ("B") "Backup"
 ("C") "Copy"
 ("D") "Delete"
 ("E") "Edit"
 ("F") "Forward"
 ("I") "Ignore"
 ("IN") "Information Only set/remove"
 ("H") "Headerless Print"
 ("K") "Priority replies set/remove"
 ("L") "Later"
 ("N") "New"
 ("P") "Print"
 ("Q") "Query"
 ("QR") "Query Recipients"
 ("QD") "Query Detailed"
 ("QN") "Query Network"
 ("R") "Reply"
 ("S") "Save"
 ("T") "Terminate"
 ("V") "Vaporize date edit"
 ("W") "Write"
 ("X") "Xtract KIDS/PackMan"

15.0 Servers—Message Activities

15.1.1 ^XMA1C

15.1.1.1 REMSBMSG^XMA1C: Delete a Message from a Server Basket

Reference Type	Supported
Category	Servers—Message Activities (Classic MailMan)
IA #	10072
Description	This API deletes a message from a server basket. The message is <i>not</i> put in the "WASTE" basket. All server baskets belong to the Postmaster, so it is not necessary to specify which user. Software applications should call this API when they have processed the server message, otherwise it will never be removed from the basket. The MailMan purge routines do not touch messages in server baskets.



Note: Compare this API to the ZAPSERV^XMXAPI(): Delete a Message from a Server Basket API described in Chapter 4, "Message Actions," in this manual.

Format REMSBMSG^XMA1C

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMSE: (required) Server name. Must be the full name, starting with "S." XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None
Variables KILLED Upon Exit	XMKD, XMZ, XMDUZ, XMK, XMSE

15.1.1.2 SETSB^XMA1C: Put a Message in a Server Basket

Reference Type	Supported
Category	Servers—Message Activities (Classic MailMan)
IA #	10072

Description This API puts a message in a server basket. All server baskets belong to the Postmaster, so it is not necessary to specify which user. Generally, software applications will not be using this call, because MailMan delivers server messages to server baskets.



Note: Compare this API to the PUTSERV^XMXAPI(): Put a Message in a Server Basket API described in Chapter 4, "Message Actions," in this manual.

Format SETSB^XMA1C

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMXX: (required) Server name. Must be the full name, starting with "S." XMZ: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output Variables	None
Variables KILLED Upon Exit	None

15.1.2 ^XMS1

15.1.2.1 \$\$SRVTIME^XMS1(): Set Server-related Fields in the Message File

Reference Type	Supported
Category	Servers—Message Activities (Classic MailMan)
IA #	1151
Description	This extrinsic function sets the LAST READ DATE/TIME field (#2) (#2):MESSAGE File (#3.9) and STATUS field (#5) (#5):MESSAGE File (#3.9) of a (server) message recipient in the RECIPIENT Multiple field of a message in the MESSAGE file (#3.9). Field #2 is set to the current date/time. Field #5 is set to the STATUS parameter.



Note: There is no other equivalent API.

Format	\$\$SRVTIME^XMS1(xmz,xmser,status)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmser: (required) Server name. Must be the full name, starting with "S." status: (required) Status string to put in the STATUS field.

Output returns:
Returns:
0—No error.
"1 No Update"—If XMSER not found in the RECIPIENT multiple.
"2 Status too long"—If STATUS is longer than 30 characters.
"3 Bad Characters in Status"—If STATUS contains a caret ("^").

15.1.2.2 \$\$STATUS^XMS1(): Get Status of a Server Recipient

Reference Type Supported
Category Servers—Message Activities (Classic MailMan)
IA # 1151
Description This extrinsic function returns the STATUS field (#5) of a (server) recipient from the RECIPIENT multiple of a message in the MESSAGE file (#3.9). If the recipient *cannot* be found, it returns null.



Note: There is no other equivalent API.

Format \$\$STATUS^XMS1(xmz,xmser)
Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
xmser: (required) Server name. Must be the full name, starting with "S."
Output returns:
Returns:
Successful—STATUS field (#5) value in the MESSAGE file (#3.9)
Unsuccessful—Null, recipient could *not* be found

16.0 Utilities—General Development

16.1.1 ^XM

16.1.1.1 ^XM: Direct Entry Into MailMan (Without Menus)

Reference Type	Supported
Category	Utilities—General Development (Classic MailMan)
IA #	10064
Description	This API is the main programmer entry point into MailMan. It is meant to be used by a programmer to enter MailMan without going through the menu system. It gives a programmer access to many MailMan options..



Note: This is a self-contained entry point, and it needs no other calls.

Format ^XM

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	DUZ: (required) User DUZ.
Output Variables	None

16.1.1.2 KILL^XM: MailMan Variable Cleanup

Reference Type	Supported
Category	Utilities—General Development (Classic MailMan)
IA #	10064
Description	This API KILLS any MailMan variables that may be left over from previous calls.
Format	KILL^XM
Input Variables	None
Output Variables	None

16.1.1.3 N1^XM: Create a Mailbox for a User

Reference Type	Supported
Category	Utilities—General Development (Classic MailMan)
IA #	10064

Description This API creates a mailbox for a user.



REF: Compare to the NEW^XM: API in this chapter and the CRE8MBOX^XMXAPIB(): Create a Mailbox API described in Chapter 7, "Basket Actions," in this manual.

Format N1^XM

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables XMDUZ: (required) User's DUZ.
 XMZ: (optional) If you wish to prevent the user from seeing messages created before a certain date, then set XMZ to a message number in the MESSAGE file (#3.9). The user will not be able to access any messages created earlier than this one, unless the message is already in the user's mailbox or is forwarded to the user. This really only applies to users who left the organization and then returned, or if (heaven forbid) you are re-using a DUZ. This prevents the user from accessing old messages that may have been addressed to the user.

Output Variables None

16.1.1.4 NEW^XM: Create a Mailbox for a User

Reference Type Supported
Category Utilities—General Development (Classic MailMan)
IA # 10064
Description This API create a mailbox for a user.



REF: Compare to the N1^XM: API in this chapter and the CRE8MBOX^XMXAPIB(): Create a Mailbox API described in Chapter 7, "Basket Actions," in this manual.

Format NEW^XM

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMZ: (optional) If you wish to prevent the user from seeing messages created before a certain date, then set XMZ to a message number in the MESSAGE file (#3.9). The user will not be able to access any messages created earlier than this one, unless the message is already in the user's mailbox or is forwarded to the user. This really only applies to users who left the organization and then returned, or if (heaven forbid) you are re-using a DUZ. This prevents the user from accessing old messages that may have been addressed to the user. Y: (required) User's DUZ.
Output Variables	None

16.1.2 ^XMADGO

16.1.2.1 ZTSK^XMADGO: Start Tasks to Deliver Messages in Local Delivery Queues

Reference Type	Supported
Category	Utilities—General Development (Classic MailMan)
IA #	10068
Description	This API starts tasks to deliver messages in local delivery queues.
Format	ZTSK^XMADGO
Input Variables	None
Output Variables	None

16.1.3 ^XMCTLK

16.1.3.1 GO^XMCTLK: Display Keyboard & Data Entries (Interactive)

Reference Type	Supported
Category	Utilities—General Development (Classic MailMan)
IA #	1148
Description	This API lets you interactively use a device and displays keyboard entry and data coming down the line. It is good for testing devices, network outgoing points, etc. The VistA programming environment is assumed (initialized through D ^XUP or signon through ^XUS). All I/O from the keyboard and device chosen are echoed on the screen. What is displayed on the screen can be captured into a mail message. Type an "A" to communicate with TalkMan.
Format	GO^XMCTLK
Input Variables	None
Output Variables	None

16.1.4 ^XMCU1

16.1.4.1 \$\$DECODEUP^XMCU1(): Convert ~U~ to ^ in a String

Reference Type	Supported
-----------------------	-----------

Category Utilities—General Development (Classic MailMan)
IA # 1136
Description This extrinsic function takes a string, converts any ~U~ to ^, and returns the result.



REF: This API is identical to the \$\$DECODEUP^XMXUTIL1(): API described in Chapter 18, "Utilities—Dates and Strings," in this manual.

Format \$\$DECODEUP^XMCU1(string)
Input Parameters string: (required) Any character string.
Output returns: Returns the input string with all ~U~ converted to carets ("^").

16.1.4.2 \$\$ENCODEUP^XMCU1(): Convert ^ to ~U~ in a String

Reference Type Supported
Category Utilities—General Development (Classic MailMan)
IA # 1136
Description This extrinsic function takes a string, converts any ^ to ~U~, and returns the result.



REF: This API is identical to the \$\$ENCODEUP^XMXUTIL1(): API described in Chapter 18, "Utilities—Dates and Strings," in this manual.

Format \$\$ENCODEUP^XMCU1(string)
Input Parameters string: (required) Any character string.
Output returns: Returns the input string with all carets ("^") converted to ~U~.

16.1.4.3 \$\$RTRAN^XMCU1(): Undo \$\$STRAN^XMCU1 Conversion

Reference Type Supported
Category Utilities—General Development (Classic MailMan)
IA # 1136
Description This extrinsic function takes a string that had been converted by \$\$STRAN^XMCU1(): , undoes the conversion, and returns the result. Printable characters are converted back into control characters.
Format \$\$RTRAN^XMCU1(string)
Input Parameters string: (required) Any character string converted by \$\$STRAN^XMCU1.
Output returns: Returns unconverted string.

16.1.4.4 \$\$STRAN^XMCU1(): Convert Control Characters to Printable Characters in a String

Reference Type Supported
Category Utilities—General Development (Classic MailMan)
IA # 1136
Description This extrinsic function takes a string, converts any control characters to printable characters, and returns the result. The conversion can be undone by the \$\$RTRAN^XMCU1 API.
Format \$\$STRAN^XMCU1(string)
Input Parameters string: (required) Any character string.

Output returns: Returns a character sting with control characters converted into printable characters.

16.1.5 ^XMUT7

16.1.5.1 ENT^XMUT7(): Send a Test Message to a User's Forwarding Address

Reference Type Supported

Category Utilities—General Development (Classic MailMan)

IA # 1132

Description This API sends a test message to a user's forwarding address. The message is also sent to the Postmaster. If the forwarding address is no good, the Postmaster receives an error message.

Format ENT^XMUT7(y)

Input Parameters y: (required) DUZ of user, whose forwarding address you want to test.

Output returns: Results: Successful—Test message gets sent to the user's forwarding address and the Postmaster. Unsuccessful—An error message is sent to the Postmaster.

17.0 Utilities—Messages and Mailboxes

17.1 ^XMXUTIL

17.1.1 \$\$BMSGCT^XMXUTIL(): Get the Number of Messages in a User's Basket

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the number of messages in a user's basket.
Format	\$\$BMSGCT^XMXUTIL(xmduz,xmk)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN.
Output	returns: Returns the number of messages in a user's basket.

17.1.2 \$\$BNMSGCT^XMXUTIL(): Get the Number of New Messages in a User's Basket

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the number of messages in a user's basket.
Format	\$\$BNMSGCT^XMXUTIL(xmduz,xmk)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN.
Output	returns: Returns the number of messages in a user's basket.

17.1.3 \$\$BPMSGCT^XMXUTIL(): Get the Number of New Priority Messages in a User's Basket

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the number of new priority messages in a user's basket.
Format	\$\$BPMSGCT^XMXUTIL(xmduz,xmk)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN.
Output	returns: Returns the number of messages in a user's basket.

17.1.4 \$\$BSKTNAME^XMXUTIL(): Get the Name of a User's Basket

Reference Type	Supported
-----------------------	-----------

Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the name of a user's basket.
Format	\$\$BSKTNAME^XMXUTIL(xmduz,xmk)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN.
Output	returns: Returns the number of a user's basket.

17.1.5 KVAPOR^XMXUTIL(): Set/Remove a Message Vaporize Date in a User's Basket

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This API sets/removes a message vaporize date in a user's basket.
Format	KVAPOR^XMXUTIL(xmduz,xmk,xmz,xmvapor,.xmiu)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmvapor: (required) Date/time (in VA FileMan format) to delete this message from this user's basket: ="@" (at-sign) to remove the vaporize date.
Output Parameters	.xmiu: User information, as related to the message: ("KVAPOR") Set to XMVAPOR or KILLED if XMVAPOR="@"

17.1.6 LASTACC^XMXUTIL(): Record that the User has Read the Message

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This API records that the user has read the message. This routine needs to be called only by those applications that display, using their own routines, messages and responses to the user. This routine sets the first and last times that the user has read the message. It records the last response that the user has read. If the user is a surrogate, it records the surrogate was the last reader. If MailMan had set a vaporize date for the message in the user's basket (because the user had not accessed it in a while), then that vaporize date is deleted. It also sends a confirmation message to the sender, if one was requested, the first time the user reads the message.
Format	LASTACC^XMXUTIL(xmduz,xmk,xmz,xmresp,.xmim,.xminstr,.xmiu,.xmconfrm)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmresp: (required) Last response read by the user this time. .xmim: (required) Message information, set by INMSG1^XMXUTIL2: ("SUBJ") Subject. ("FROM") Sender.

.xminstr: (required) More message information, set by INMSG2^XMXUTIL2: ("FLAGS") Special instructions (here, we are interested in whether "FLAGS"["R"—confirm receipt requested).

.xmiu: (required) User information, as related to the message ("IEN") IEN of user record in message RECIPIENT multiple, set by INMSG1^XMXUTIL2.

("RESP") Last response read by the user, initially set by INMSG1^XMXUTIL2 or INRESPS^XMXUTIL2.

Output

.xmiu: User information, as related to the message:

("RESP") If XMRESP is greater than XMIU("RESP"), then XMIU("RESP") is set to XMRESP.

.xmconfirm: Was a confirmation message sent to the message sender?
0—No 1—Yes

17.1.7 MAKENEW^XMXUTIL(): Make a Message New & Update the New Message Counts

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This API makes a message new and updates the new message counts.
Format	MAKENEW^XMXUTIL(xmduz,xmk,xmz[,xmlockit])
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmlockit: (optional) Should MailMan take care of locking and unlocking the ^XMB(3.7,XMDUZ global? 0 (default)—No 1—Yes



NOTE: The locking *must* be done to ensure the integrity of the new message counts. If MailMan does not do it, then the calling application must.

Output None

17.1.8 \$\$NAME^XMXUTIL(): Get the User's Name, Title, and/or Institution

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the name of the user by looking up XMDUZ in the NEW PERSON file (#200). Optionally, it can also return the user's Title and/or Institution. If XMDUZ is not numeric, it returns XMDUZ.
Format	\$\$NAME^XMXUTIL(xmduz[,xminfo])
Input Parameters	xmduz: (required) User DUZ.

xminfo: (optional) If the variables XMV("SHOW INST") and XMV("SHOW TITL") indicate that the user's Institution and/or Title are desired, should that information be returned, too?
 0 (default)—No
 1—Yes

Output returns: Returns the user's Name, Title, and/or Institution.

17.1.9 \$\$NETNAME^XMXUTIL(): Get User's Network Name & Domain

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This extrinsic function returns network name of user, including @site name.
Format \$\$NETNAME^XMXUTIL(xmduz)
Input Parameters xmduz: (required) User DUZ or any string.
Output returns: Returns the user's Network Name and Domain name (i.e., @site name).

17.1.10 \$\$NEWS^XMXUTIL(): Get Information on New Messages in a User's Mailbox

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This extrinsic function returns information on new messages in a user's mailbox. This function returns much the same information as the routine QMBOX^XMXAPI.
Format \$\$NEWS^XMXUTIL(xmduz[,xmtest])
Input Parameters xmduz: (required) User DUZ.
 xmtest: (optional) Is this a test? 1 (default)—Yes 0—No If this is *not* a test, then the LAST NEW MSG NOTIFY DATE/TIME field (#1.12) in the MAILBOX file (#3.7) can be updated for this user.
Output returns: Returns:
 -1—If XMDUZ is not a valid user
 0—If the user has no new messages
 Otherwise, it returns the following ^-delimited string:
 Piece 1: Number of new messages in the mailbox.
 Piece 2: Does the user have new priority mail? 0—No 1—Yes
 Piece 3: Number of new messages in the "IN" basket
 Piece 4: Date/time (in VA FileMan format) that the last message was received.
 Piece 5: Have there been any new messages since the last time this routine was called? 0—No 1—Yes

17.1.11 NONEW^XMXUTIL(): Make a Message *Not* New & Update the New Message Counts

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This API makes a message *not* new and updates the new message counts.
Format NONEW^XMXUTIL(xmduz,xmk,xmz,xmlockit)

Input Parameters xmduz: (required) User DUZ.
 xmk: (required) Basket IEN.
 xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xmlockit: (optional) Should MailMan take care of locking and unlocking the
 ^XMB(3.7,XMDUZ global? 0 (default)—No 1—Yes.



NOTE: The locking *must* be done to ensure the integrity of the new message counts. If MailMan does not do it, then the calling application must.

Output None

17.1.12 PAGE^XMXUTIL(): Display "Continue" Prompt to User

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This API displays to the user: "Enter RETURN to continue or '^' to exit:" and waits until the user presses a key. It sets up and uses the standard VA FileMan call to do this.
Format PAGE^XMXUTIL(.xmabort)
Input Parameters None
Output Parameters .xmabort: Did the user choose to exit? 0—No 1—Yes

17.1.13 \$\$TMSGCT^XMXUTIL(): Get the Total Number of Messages in a User's Mailbox

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This extrinsic function returns the total number of messages in a user's mailbox.
Format \$\$TMSGCT^XMXUTIL(xmduz)
Input Parameters xmduz: (required) User DUZ.
Output returns: Returns the total number of messages in a user's mailbox.

17.1.14 \$\$TNMSGCT^XMXUTIL(): Get the Total Number of New Messages in a User's Mailbox

Reference Type Supported
Category Utilities—Messages and Mailboxes
IA # 2734
Description This extrinsic function returns the total number of new messages in a user's mailbox.
Format \$\$TNMSGCT^XMXUTIL(xmduz)
Input Parameters xmduz: (required) User DUZ.
Output returns: Returns the total number of new messages in a user's mailbox.

17.1.15 \$\$TPMSGCT^XMXUTIL(): Get the Total Number of New Priority Messages in a User's Mailbox

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This extrinsic function returns the total number of new priority messages in a user's mailbox.
Format	\$\$TPMSGCT^XMXUTIL(xmduz)
Input Parameters	xmduz: (required) User DUZ.
Output	returns: Returns the total number of new priority messages in a user's mailbox.

17.1.16 WAIT^XMXUTIL(): Display "Continue" Prompt to User

Reference Type	Supported
Category	Utilities—Messages and Mailboxes
IA #	2734
Description	This API displays to the user: "Press RETURN to continue:" and waits until the user presses a key. Sets up and uses the standard VA FileMan call to do this.
Format	WAIT^XMXUTIL
Input Parameters	None
Output	None

18.0 Utilities—Dates and Strings

18.1.1 ^XMXUTIL1

18.1.1.1 \$\$CONVERT^XMXUTIL1(): Convert Internet Date/Time to VA FileMan Date/Time

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function converts an Internet Date/Time string into a VA FileMan Date/Time. If the Internet Date/Time string <i>cannot</i> be understood, it returns -1.
Format	\$\$CONVERT^XMXUTIL1(x,xmtime)
Input Parameters	x: (required) Internet Date/Time string. xmtime: (optional) Should the time also be converted? 0 (default)—No. If no, it returns the VA FileMan date only 1—Yes.
Output	returns: Returns: Successful—VA FileMan Date/Time converted string. Unsuccessful—If the Internet Date/Time input string <i>cannot</i> be understood, it returns -1.

18.1.1.2 \$\$CTRL^XMXUTIL1(): Strip Control Characters from a String

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function strips control characters from a string.
Format	\$\$CTRL^XMXUTIL1(xmstring)
Input Parameters	xmstring: (required) The input string.
Output	returns: Returns the input string without any control characters.

18.1.1.3 \$\$DECODEUP^XMXUTIL1(): Convert All ~U~ to ^ in a String

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function converts all ~U~ to a caret ("^") in a string.



NOTE: See also: \$\$DECODEUP^XMCU1(): in Chapter 16, "Utilities—General Development," in this manual.

Format	\$\$DECODEUP^XMXUTIL1(xmstring)
Input Parameters	xmstring: (required) The input string.
Output	returns: Returns the input string with all ~U~ converted to a caret ("^").

18.1.1.4 \$\$ENCODEUP^XMXUTIL1(): Convert All ^ to ~U~ in a String

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function converts all ~U~ to a caret ("^") in a string.



NOTE: See also: \$\$DECODEUP^XMCU1(): in Chapter 16, "Utilities—General Development," in this manual.

Format	\$\$ENCODEUP^XMXUTIL1(xmstring)
Input Parameters	xmstring: (required) The input string.
Output	returns: Returns the input string with all carets ("^") converted to ~U~.

18.1.1.5 \$\$GMTDIFF^XMXUTIL1(): Get the +-hhmm Difference from Greenwich Mean Time (GMT)

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function returns the +-hhmm difference from Greenwich Mean Time (GMT) given the time zone. If there's no record of the time zone, it returns the null string.
Format	\$\$GMTDIFF^XMXUTIL1(xmzone)
Input Parameters	xmzone: (required) The 3-character time zone.
Output	returns: Returns: Successful—The +-hhmm difference from Greenwich Mean Time (GMT). Unsuccessful—Null string.

18.1.1.6 \$\$INDT^XMXUTIL1(): Convert VA FileMan Date/Time to Internet Date/Time

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function converts VA FileMan Date/Time into an Internet Date/Time string: dd mm yy hh:mm:ss +-hhmm (time zone)
Format	\$\$INDT^XMXUTIL1(xmdt)
Input Parameters	xmdt: (required) VA FileMan Date/Time.
Output	returns: Returns the Internet Date/Time converted string.

18.1.1.7 \$\$MAXBLANK^XMXUTIL1(): Reduce Consecutive Spaces in a String

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function reduces all three or more consecutive spaces in a string to two spaces.
Format	\$\$MAXBLANK^XMXUTIL1(xmstring)

Input Parameters xmstring: (required) The input string.
Output returns: Returns the input string with all three or more consecutive spaces reduced to two spaces.

18.1.1.8 \$\$MELD^XMXUTIL1(): Combine a String & Number to Form a New String of a Given Length

Reference Type Supported
Category Utilities—Dates and Strings
IA # 2735
Description This extrinsic function combines a string and a number to form a new string of a given length. The string will be right justified; the number left-justified, with at least two spaces separating the string and number. The string will be truncated, if necessary.
Format \$\$MELD^XMXUTIL1(xmstring[,xmnumber],xmlen)
Input Parameters xmstring: (required) The input string.
xmnumber: (optional) The number.
xmlen: (required) The length of the new string to be formed.
Output returns: Returns the newly formed string.

Example 1

```
>W $$MELD^XMXUTIL1("Lotus blossom",123,10)
Lotus 123
```

Example 2

```
>W $$MELD^XMXUTIL1("Lotus blossom",123,15)
Lotus blos 123
```

18.1.1.9 \$\$MMDT^XMXUTIL1(): Reformat VA FileMan Date

Reference Type Supported
Category Utilities—Dates and Strings
IA # 2735
Description This extrinsic takes a VA FileMan Date/Time input string and returns it as a reformatted string: mm/dd/yy@hh:mm
Format \$\$MMDT^XMXUTIL1(xmdt)
Input Parameters xmdt: (required) VA FileMan Date/Time input string.
Output returns: Returns a reformatted VA FileMan Date/Time string: mm/dd/yy@hh:mm

Example

```
>W $$MMDT^XMXUTIL1(2940629.105744)
06/29/94@10:57
```

18.1.1.10 \$\$SCRUB^XMXUTIL1(): Strip Control Characters & Leading/Trailing Spaces from a String.

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function strips control characters and leading/trailing spaces from a string.
Format	\$\$SCRUB^XMXUTIL1(xmstring)
Input Parameters	xmstring: (required) The input string.
Output	returns: Returns the input string stripped of any control characters and/or leading/trailing spaces.

18.1.1.11 \$\$STRIP^XMXUTIL1(): Strip Leading/Trailing Spaces from a String

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function strips leading/trailing spaces from a string.
Format	\$\$STRIP^XMXUTIL1(xmstring)
Input Parameters	xmstring: (required) The input string.
Output	returns: Returns the input string stripped of any leading/trailing spaces.

18.1.1.12 \$\$TIMEDIFF^XMXUTIL1(): Reformat Decimal Time Difference to +-hhmm

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function returns +-hhmm when given the decimal time difference (between time zones).
Format	\$\$TIMEDIFF^XMXUTIL1(xmdiff)
Input Parameters	xmdiff: (required) Decimal time difference.
Output	returns: Returns reformatted decimal time difference to +-hhmm.

Example

```
>W $$TIMEDIFF^XMXUTIL1(-2.5)
-0230
```

18.1.1.13 \$\$TSTAMP^XMXUTIL1: Get a Timestamp (\$H Expressed in Seconds)

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This extrinsic function returns a timestamp (\$H expressed in seconds).
Format	\$\$TSTAMP^XMXUTIL1
Input Parameters	None
Output	returns: Returns the current timestamp (i.e., \$H expressed in seconds).

Example

```
>W $$TSTAMP^XMXUTIL1
5229582368
```

18.1.1.14 ZONEDIFF^XMXUTIL1(): Get Time Difference Between Time Zone and Local Time

Reference Type	Supported
Category	Utilities—Dates and Strings
IA #	2735
Description	This API, given the time zone (or time difference +hhmm from Greenwich Mean Time [GMT]), returns the number of hours and minutes difference between the input and the local time zone.
Format	ZONEDIFF^XMXUTIL1(xmyt,.xmhh,.xmmm)
Input Parameters	xmyt: (required) Time zone (3-character or +-hhmm from GMT).
Output Parameters	.xmhh: Number of hours time difference. .xmmm: Additional number of minutes time difference.

19.0 Utilities—Message Information

These APIs retrieve the following categories of information about a message:

- Information to be displayed.
- Information used to determine what can (and *cannot*) be done with the message.

19.1 ^XMXUTIL2

19.1.1 \$\$BSKT^XMXUTIL2(): Get Basket Information

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns which basket a message is in for a user. It returns the following: 0—Not in a basket for this user Number—It's in this basket IEN for the user. (XMNAME=0) Number^name—It's in this basket IEN of this name for the user. (XMNAME=1)
Format	\$\$BSKT^XMXUTIL2(xmduz,xmz[,xmname])
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmname: (optional) Return the basket name, too? 0 (default)—No 1—Yes
Output	returns: Returns: 0—Not in a basket for this user Number—It's in this basket IEN for the user. (XMNAME=0) Number^name—It's in this basket IEN of this name for the user. (XMNAME=1)

19.1.2 \$\$DATE^XMXUTIL2(): Get Message Sent Date

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns the message sent date. It is returned in external format: DD MMM YY HH:MM



NOTE: Compare this API to the \$\$ZDATE^XMXUTIL2(): API described in this chapter.

Format	\$\$DATE^XMXUTIL2(xmzrec[,xmtime])
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0). xmtime: (optional) Return the time, also? 1 (default)—Yes, date and time 0—No, date only
Output	returns: Returns message sent date in external date format: DD MMM YY HH:MM

19.1.3 \$\$FROM^XMXUTIL2(): Get Message From Information

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns the message sent date. It is returned in external format.



NOTE: Compare this API to the \$\$ZDATE^XMXUTIL2(): API described in this chapter.

Format	\$\$FROM^XMXUTIL2(xmzrec)
Input Parameters	xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	returns: Returns message sent date in external date format

19.1.4 INMSG^XMXUTIL2(): Get Message Information

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736



NOTE: This API should only be called for messages, not for responses. This routine calls both the INMSG1^XMXUTIL2 and INMSG2^XMXUTIL2 APIs. It also returns additional information.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API in Chapter 5, "Getting Information About and Text From Messages," in this manual.

Format	INMSG^XMXUTIL2(xmduz,xmk,xmz[,xmzrec][,xmflags,].xmim,.xminstr,.xmiu)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN. (Set XMK=0, if the message is <i>not</i> in a basket or if you are <i>not</i> interested in variables XMIU("KVAPOR") and XMIU("NEW").) xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). Xmzrec (optional) Zero node of the message: ^XMB(3.9,XMZ,0). xmflags: (optional) Used to control output: I—Internal values only. The default is internal values, and, where it makes sense, to set variables with other values, too. F—Set variable with internal VA FileMan date format. The default is external MailMan date format. "F" is ignored if XMFLAGS contains "I".
Output	.xmim: Message information (KILLED first). For a description of this parameter, please refer to the definition of XMIM for INMSG1^XMXUTIL2 (described below): "SUBJ", "ENV FROM", "FROM", "FROM DUZ", "FROM NAME", "DATE", "DATE FM", "DATE MM", "SENDER", "SENDER DUZ", "SENDER NAME", "LINES", "RESPS", "RECIPS", "CRE8", "CRE8 MM" .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "FLAGS", "TYPE", "VAPOR", "RCPT BSKT", "SCR HINT"

.xmiu: User information, as related to the message. For a description of this parameter, please refer to the definition of XMIU for INMSG1^XMXUTIL2 and INMSG2^XMXUTIL2 (described in this chapter):
 "IEN", "RESP", "ORIGN8" ("KVAPOR") DATE/TIME (in VA FileMan format) to delete this message from this user's basket. (Set only if applicable.)
 ("NEW") Is message new? 0—No 1—Yes
 2—Yes, and priority too

The following table compares the variables returned by \$\$HDR^XMGAPI2 and INMSG^XMXUTIL2:

Table 19-1. Comparison of variables returned by \$\$HDR^XMGAPI2 and INMSG^XMXUTIL2

\$\$HDR^XMGAPI2	INMSG^XMXUTIL2
L("BLOBCNT")	N/A
L("BROADCAST")	N/A (use \$\$BCAST^XMXSEC)
L("BSKT IEN")	N/A (use \$\$BSKT^XMXUTIL2)
L("BSKT")	N/A (use \$\$BSKT^XMXUTIL2)
L("DATE FM")	XMIM("DATE FM")—If XMFLAGS["F" and XMFLAGS["I".
L("DATE")	XMIM("DATE")—Internal.
L("LINES")	XMIM("LINES")
L("NEW")	XMIU("NEW")
L("PXMZ")	N/A
L("RRCV")	XMIM("RESPTS")
L("RRED")	XMIU("RESP")
L("RSP",i)	N/A (use INRESP^XMXUTIL2 to get response information)
L("SENDER DUZ")	XMIM("FROM DUZ")—If XMFLAGS["I".
L("SENDER")	XMIM("FROM NAME")—If XMFLAGS["I".
L("SUBJ")	XMIM("SUBJ")
L("SURROG")	XMIM("SENDER NAME")—If XMFLAGS["I".
L("TYPE")	XMINSTR("TYPE")
L("XMZ")	XMIM("XMZ")
N/A	XMIM("CRE8")—Local create date.
N/A	XMIM("CRE8 MM")—MailMan external formatted date, if XMFLAGS["F" or "I".
N/A	XMIM("DATE MM")—MailMan external formatted date, if XMFLAGS["F" or "I".
N/A	XMIM("ENV FROM")—Message envelope "MAIL FROM:"
N/A	XMIM("FROM")—Internal.
N/A	XMIM("RECIPS")—Number of recipients.
N/A	XMIM("SENDER DUZ")—If XMFLAGS["I".
N/A	XMIM("SENDER")—Internal.
N/A	XMINSTR("FLAGS")—Closed, Confidential, Information Only, Priority, Confirmation requested, Priority responses.
N/A	XMINSTR("RCPT BSKT")—Delivery basket.
N/A	XMINSTR("SCR HINT")—Scramble hint.

\$\$HDR^XMGAPI2	INMSG^XMXUTIL2
N/A	XMINSTR("VAPOR")—Vaporize date of message.
N/A	XMIU("IEN")—User's IEN in RECIPIENT multiple.
N/A	XMIU("KVAPOR")—Vaporize date of message in user's basket.
N/A	XMIU("ORIGN8")—Did user send message?

19.1.5 INMSG1^XMXUTIL2(): Get Message Information (Part 1)

Reference Type Supported
Category Utilities—Message Information
IA # 2736



NOTE: This API should only be called for messages, not for responses. It calls the INRESPTS^XMXUTIL2(): Get Message Response Information API, also described in this chapter.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API in Chapter 5, "Getting Information About and Text From Messages," in this manual.

Format INMSG1^XMXUTIL2(xmduz,xmz[,xmzrec][,xmflags],.xmim,.xmiu)
Input Parameters xmduz: (required) User DUZ.
xmz: (required) IEN in the MESSAGE file (#3.9).
xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
xmflags: (optional) Used to control setting of output variables: I—Internal values only. The default is internal values, and, where it makes sense, to set variables with other values, too.) F—Set variable with internal VA FileMan date format. The default is external MailMan date format. "F" is ignored if XMFLAGS contains "I".

The Output is shown in the following table.

Output	Description	Meaning
xmim:	Message information (KILLED first):	
	("XMZ")	Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
	("SUBJ")	Subject of message (all ~U~ translated to ^).
	("ENV FROM")	"MAIL FROM:", as stated in the message envelope on a message that was received from a remote site. (Not set, if it does not exist.)
	("FROM")	Who sent the message (internal).
	("FROM DUZ")	DUZ of person who sent the message (if applicable). (Not set if XMFLAGS contains "I".)

Output	Description	Meaning
	("FROM NAME")	Name of person who sent the message. (<i>Not set if XMFLAGS contains "I".</i>)
	("DATE")	When the message was sent (internal).
	("DATE FM")	VA FileMan date (-1, if error). (Set if XMFLAGS contains "F". <i>Not set if XMFLAGS contains "I".</i>)
	("DATE MM")	External MailMan format: dd mmm yy hh:mm (Internet date, if error.) (<i>Not set, if XMFLAGS contains "I" or "F".</i>)
	("CRE8")	Local create date (in VA FileMan format).
	("CRE8 MM")	External MailMan format: dd mmm yy hh:mm (Internet date, if error.) (<i>Not set, if XMFLAGS contains "I" or "F".</i>)
	("SENDER")	Who really sent the message (if applicable).
	("SENDER DUZ")	DUZ of person who really sent the message (if applicable). <i>Not set, if XMFLAGS contains "I".</i>
	("LINES")	How many lines are in the message.
	("RESPS")	How many responses does the message have.
.xmiu:	User information, as related to the message (KILLED first).	
	("IEN")	IEN of XMDUZ in the message's RECIPIENT multiple.
	("RESP")	Number of the last response that the user has read.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API described in Chapter 5, "Getting Information About and Text From Messages," in this manual.

19.1.6 INMSG2^XMXUTIL2(): Get Message Information (Part 2)

Reference Type Supported
Category Utilities—Message Information
IA # 2736



NOTE: This API should only be called for messages, not for responses.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API in Chapter 5, "Getting Information About and Text From Messages," in this manual.

Format	INMSG2^XMXUTIL2(xmduz,xmz[,xmzrec],.xmim,.xminstr,.xmiu)
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
Output	.xmim: Message information. ("RECIPS") Number of recipients of the message. .xminstr: Special instructions on the message. For a description of this parameter, please refer to the "Parameter Definitions" topic in Chapter 4, "Message Actions," in this manual: "FLAGS", "TYPE", "VAPOR", "RCPT BSKT", "SCR HINT" .xmiu: User information, as related to the message: ("ORIGN8") Did the user send the message? 0—No 1—Yes (results from a call to \$\$ORIGIN8R^XMXSEC)



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API described in Chapter 5, "Getting Information About and Text From Messages," in this manual.

19.1.7 INRESP^XMXUTIL2: Get Response Information

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736



NOTE: This API should only be called for responses, not for messages.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API in Chapter 5, "Getting Information About and Text From Messages," in this manual.

Format	INRESP^XMXUTIL2(xmz,xmwhich[,xmflags],.xmir)
---------------	--

Make sure to perform the following steps before calling this API:

- NEW all of the input and output variables..
- Set the input variables you want changed.
- Call the API.

If you do not follow these steps, the variables could unintentionally assume the values of the variables of the current running task.

Input Variables	XMDUZ: (required) User DUZ.
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmwhich: (required) The number of the response for which to get the

information.

xmflags: (optional) Used to control output:

I—Internal values only. The default is internal values, and, where it makes sense, to set variables with other values, too.

F—Set variable with internal VA FileMan date format. The default is external MailMan date format. "F" is ignored if XMFLAGS contains "I".

Output Parameters .xmim Response information (KILLED first). For a description of this parameter, please refer to the definition of XMIM for INMSG1^XMXUTIL2(): Get Message Information (Part 1) (described in this chapter):
"XMZ", "SUBJ", "ENV FROM", "FROM", "FROM DUZ", "FROM NAME",
"DATE", "DATE FM", "DATE MM", "SENDER", "SENDER DUZ", "SENDER NAME",
"LINES"



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API described in Chapter 5, "Getting Information About and Text From Messages," in this manual.

19.1.8 INRESPS^XMXUTIL2(): Get Message Response Information

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This API returns the number of responses to a message and the number of the last response that the user has read.



REF: See also: \$\$HDR^XMGAPI2(): Set up an Array Containing Message Information API in Chapter 5, "Getting Information About and Text From Messages," in this manual.

Format	INRESPS^XMXUTIL2(xmz,.xmim,.xmiu)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). .xmiu("IEN"): (required) Set by INMSG1^XMXUTIL2.
Output	.xmim("RESPS"): Number of responses for a message. .xmiu("RESP"): Number of the last response that the user has read.

19.1.9 \$\$KSEQN^XMXUTIL2(): Get Message Sequence Number

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns the sequence number for a specific message in a specified user's basket.
Format	\$\$KSEQN^XMXUTIL2(xmduz,xmk,xmz)
Input Parameters	xmduz: (required) User DUZ. xmk: (required) Basket IEN. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output returns: Returns the sequence number for the specific message in the specified user's basket.

19.1.10 \$\$LINE^XMXUTIL2(): Get Number of Text Lines in a Message

Reference Type Supported
Category Utilities—Message Information
IA # 2736
Description This extrinsic function returns the number of lines in the text of a message.
Format \$\$LINE^XMXUTIL2(xmz)
Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output returns: Returns the number of lines in the text of the specified message.

19.1.11 \$\$NEW^XMXUTIL2(): Get New Message Indicator

Reference Type Supported
Category Utilities—Message Information
IA # 2736
Description This extrinsic function returns a value indicating whether or not a message is new for this user in this basket.
Format \$\$NEW^XMXUTIL2(xmduz,xmk,xmz)
Input Parameters xmduz: (required) User DUZ.
xmk: (required) Basket IEN.
xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output returns: Returns: 0—No 1—Yes

19.1.12 \$\$PRI^XMXUTIL2(): Get Priority Message Indicator

Reference Type Supported
Category Utilities—Message Information
IA # 2736
Description This extrinsic function returns a value indicating whether the message is priority or not .



REF: Compare this API to the \$\$ZPRI^XMXUTIL2(): API described in this chapter.

Format \$\$PRI^XMXUTIL2(xmzrec)
Input Parameters xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).
Output returns: Returns:
0—No
1—Yes

19.1.13 \$\$QRESP^XMXUTIL2(): Check if Message is a Response

Reference Type Supported
Category Utilities—Message Information
IA # 2736
Description This extrinsic function determines if a message is a response or not.

Format \$\$QRESP^XMXUTIL2(xmz[,xmzrec][,xmwhich])

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xmzrec: (optional) Zero node of the message: ^XMB(3.9,XMZ,0).
 xmwhich: (optional) If it is a response to a message, do you want to know which number response? 0 (default)—No 1—Yes

Output returns: Returns:
 0—Message xmz is *not* a response to this message.
 IEN^number—Message xmz is a response to this message:
 IEN—The Internal Entry Number (IEN) of the message in the MESSAGE file (#3.9).
 <number>—The message response number. This 2nd piece is only returned if XMWHICH=1.

19.1.14 \$\$RESP^XMXUTIL2(): Get the Number of Responses to a Message

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns the number of responses to a message.

Format \$\$RESP^XMXUTIL2(xmz)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output returns: Returns the number of responses to a message.

19.1.15 \$\$SUBJ^XMXUTIL2(): Get Message Subject

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns a value indicating whether the message is priority or not .



REF: Compare this API to the \$\$ZPRI^XMXUTIL2(): API described in this chapter.

Format \$\$SUBJ^XMXUTIL2(xmzrec)

Input Parameters xmzrec: (required) Zero node of the message: ^XMB(3.9,XMZ,0).

Output returns: Returns the message subject in external format.

19.1.16 \$\$ZDATE^XMXUTIL2(): Get Message Sent Date

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns the message sent date. It is returned in external format
 DD MMM YY HH:MM



REF: Compare this API to the \$\$DATE^XMXUTIL2(): API described in this chapter.

Format \$\$ZDATE^XMXUTIL2(xmz[,xmtime])

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
 xmtime: (optional) Return the time, also?
 1 (default)—Yes, date and time
 0—No, date only

Output returns: Returns message sent date in external format:
 DD MMM YY HH:MM

19.1.17 \$\$ZFROM^XMXUTIL2(): Get Message From Information

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns the message sent date. It is returned in external format



REF: Compare this API to the \$\$FROM^XMXUTIL2(): API described in this chapter.

Format \$\$ZFROM^XMXUTIL2(xmz)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output returns: Returns message sent date in external format.

19.1.18 \$\$ZNODE^XMXUTIL2(): Get Message Zero Node

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns the message zero node: ^XMB(3.9,XMZ,0).

Format \$\$ZNODE^XMXUTIL2(xmz)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output returns: Returns the message zero node: ^XMB(3.9,XMZ,0)

19.1.19 \$\$ZPRI^XMXUTIL2(): Get Priority Message Indicator

Reference Type Supported

Category Utilities—Message Information

IA # 2736

Description This extrinsic function returns a value indicating whether the message is priority or not .



REF: Compare this API to the \$\$PRI^XMXUTIL2(): API described in this chapter.

Format \$\$ZPRI^XMXUTIL2(xmz)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

Output returns: Returns:
 0—No, message is not priority.
 1—Yes, message is not priority

19.1.20 \$\$ZREAD^XMXUTIL2(): Get the Number of Responses Read

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns the number of responses to a message that a specified user has read.
Format	\$\$ZREAD^XMXUTIL2(xmduz,xmz)
Input Parameters	xmduz: (required) User DUZ. xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns: Null—User has not read the message at all 0—User has read the original message only Number—User has read through this response

19.1.21 \$\$ZSUBJ^XMXUTIL2(): Get Message Subject

Reference Type	Supported
Category	Utilities—Message Information
IA #	2736
Description	This extrinsic function returns the message subject. It is returned in external format.



REF: Compare this API to the \$\$\$SUBJ^XMXUTIL2(): API described in this chapter.

Format	\$\$ZSUBJ^XMXUTIL2(xmz)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).
Output	returns: Returns message sent date in external format.

19.2 ^XMXUTIL3

19.2.1 Q^XMXUTIL3(): List/Find Message Addressees

Reference Type	Supported
Category	Utilities—Message Information
IA #	2737
Description	This API returns a list of the addressees of a specified message. Optionally, it finds addressees that match a string. It gets a list of the requested addressees (similar in format to that produced by LIST^DIC).
Format	Q^XMXUTIL3(xmz[,xmflags][,xmamt][,xmstart][,xmfind],xmroot)
Input Parameters	xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9). xmflags: (optional) Reserved for future use. xmamt: (optional) How many? * (default)—Get all Number—Get this many .xmstart: (optional) Used to start the Lister. The Lister keeps it updated from call to call. ("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.

xmfind: (optional) Find the addressees that match the string. (VA FileMan's FIND^DIC is used.) If XMFIND is supplied, then the xmamt and xmstart parameters are ignored; a complete list is always returned.

xmroot: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).

Output Parameters

.xmstart: Used to start the Lister. The Lister keeps it updated from call to call.

("IEN") Start *after* this addressee IEN.

Continues from there, with each successive call, to the end. The default is to start at the beginning.

xmroot: Fields returned under XMTRoot for each addressee:

"TO NAME" Addressee name.

"TYPE" Addressee type (if present): I—Information Only C—cc (Carbon Copy)

19.2.2 QD^XMXUTIL3(): List/Find Message Recipients

Reference Type Supported

Category Utilities—Message Information

IA # 2737

Description This API returns a list of the recipients of this message. Optionally, it finds recipients that match a string. It gets a list of the requested recipients (similar in format to that produced by LIST^DIC).

Format QD^XMXUTIL3(xmz[,xmflags][,xmamt][,xmstart][,xmfind],xmroot)

Input Parameters xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).

xmflags: (optional) Reserved for future use.

xmamt: (optional) How many?

* (default)—Get all

Number—Get this many

.xmstart: (optional) Used to start the Lister. The Lister keeps it updated from call to call.

("IEN") Start *after* this addressee IEN.

Continues from there, with each successive call, to the end. The default is to start at the beginning.

xmfind: (optional) Find the recipients that match the string. (VA FileMan's FIND^DIC is used.) If XMFIND is supplied, then the xmamt and xmstart parameters are ignored; a complete list is always returned.

xmroot: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).

The Output parameters are shown in the following table:

Output Parameters	Description	Meaning
xmstart:	Used to start the Lister. The Lister keeps it updated from call to call.	("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.
xmroot:	Fields returned under XMTRoot for each recipient:	
	"TO"	Recipient .01 field (could be DUZ or text).
	"TO NAME"	Recipient name.

Output Parameters	Description	Meaning
	"TO ID"	ID of recipient: L—Local user F—Fax R—Remote S—Server D—Device *—Broadcast
	"TYPE"	(if present) Recipient type: • C—cc (Carbon Copy) • I—Information Only
	"FWD BY DUZ"	(if present) DUZ of the person who forwarded to this recipient.
	"FWD BY"	(if present) Name of the person, possibly followed by, in parentheses, the name of the surrogate of the person, who forwarded to this recipient.
	"FWD ON"	Date that message was forwarded to this recipient in MailMan format: dd mmm yy hh:mm
	"FWD TYPE"	(present only if forwarding is not "regular") Type of forwarding: • R (default)—Regular-Forward • F—Filter-Forward • A—Auto-Forward
	"FWD BY ORIG"	(present only if "FWD TYPE" is "A") Name of the person, possibly followed by, in parentheses, the name of the surrogate of the person, who forwarded the message to the recipient, who had auto-forwarding.
	Depending on "TO ID", the following fields are also returned: • "TO ID"="L"—Local User:	
	"TO DUZ"	DUZ of the local recipient.
	"RESP"	(if present) Number of the last response read (zero equals original message).
	"LREAD"	(if present) DATE/TIME (in VA FileMan format) the message was last read.
	"LREAD MM"	(if present) DATE/TIME (in MailMan format) the message was last read.
	"FREAD"	(if present) DATE/TIME (in VA FileMan format) the message was first read.
	"FREAD MM"	(if present) DATE/TIME (in MailMan format) the message was first read.
	"COPY"	(if present) DATE/TIME (in VA FileMan format) the message was last copied.
	"COPY MM"	(if present) DATE/TIME (in MailMan format) the message was last copied.

Output Parameters	Description	Meaning
	"TERM"	(if present) DATE/TIME (in VA FileMan format) the message was terminated.
	"TERM MM"	(if present) DATE/TIME (in MailMan format) the message was terminated.
	"SURRE"	(if present) Name of the surrogate who last read the message.
	• "TO ID"="*"—Broadcast:	
	No additional fields.	
	• "TO ID"="F"—Fax:	
	"FDATE"	(if present) DATE/TIME (in VA FileMan format) the message was passed to the Fax software.
	"FDATE MM"	(if present) DATE/TIME (in MailMan format) the message was passed to the Fax software.
	"STATUS"	(if present) Status of the fax (present before the message is passed to the Fax software).
	"FAX IEN"	(if present) IEN (in FAX ROLODEX file) of the fax recipient (present before the message is passed to the Fax software).
	"ID"	(if present) Fax ID (present after the message is passed to the Fax software).
	• "TO ID"="R"—Remote:	
	"XDATE"	(if present) DATE/TIME (in VA FileMan format) the transmission of the message began (present after transmission is complete).
	"XDATE MM"	(if present) DATE/TIME (in MailMan format) the transmission of the message began (present after transmission is complete).
	"STATUS"	(if present) Status of the message (present before and during transmission, and if error, afterward).
	"ID"	(if present) Message ID (present after transmission is successfully completed).
	"PATH"	(if present) IEN (in DOMAIN file [#4.2]) of the Domain to/via which the message will be sent (present before and during transmission).
	"PATH NAME"	(if present) Name of the Domain to/via which the message will be sent (present before and during transmission).
	"SECS"	(if present) Duration of the transmission (in seconds, present after transmission is complete).
	• "TO ID"="D"—Device or "S"—Server:	
	"SDATE"	(if present) DATE/TIME (in VA FileMan format) each time the STATUS changes, once a task starts dealing with the message.

Output Parameters	Description	Meaning
	"SDATE MM"	(if present) DATE/TIME (in MailMan format) each time the STATUS changes, once a task starts dealing with the message.
	"STATUS"	(if present) Status of the message (usually present before, during, and after sending).

19.2.3 QL^XMXUTIL3(): List/Find "Latered" Message Addressees

Reference Type	Supported
Category	Utilities—Message Information
IA #	2737
Description	This API returns a list of the "latered" addressees of this message. Optionally, it finds the "latered" addressees that match a string. It gets a list of the requested "latered" addressees (similar in format to that produced by LIST^DIC).
Format	QL^XMXUTIL3(xmz[,xmflags][,xmamt][,.xmstart][,xmfind],xmroot)
Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>xmflags: (optional) Reserved for future use.</p> <p>xmamt: (optional) How many? * (default)—Get all Number—Get this many</p> <p>.xmstart: (optional) Used to start the Lister. The Lister keeps it updated from call to call.</p> <p>("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.</p> <p>xmfind: (optional) Find the "latered" addressees that match the string. (VA FileMan's FIND^DIC is used.) If XMFIND is supplied, then the xmamt and xmstart parameters are ignored; a complete list is always returned.</p> <p>xmroot: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).</p>

The Output parameters are shown in the following table:

Output Parameters	Description	Meaning
xmstart:	Used to start the Lister. The Lister keeps it updated from call to call.	("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.
xmroot:	Fields returned under XMTROOT for each "latered" addressee:	
	"TO NAME"	Latered addressee name.
	"TYPE"	(if present) Addressee type: <ul style="list-style-type: none"> • I—Information Only • C—cc (Carbon Copy)
	"BY DUZ"	DUZ of the person who "latered."
	"BY NAME"	Name of the person who "latered."
	"WHEN"	When will the message be delivered (in VA FileMan format).

Output Parameters	Description	Meaning
	"WHEN MM"	When will the message be delivered (in MailMan format: dd mmm yy hh:mm)

19.2.4 QN^XMXUTIL3(): Get Network Message Header Records

Reference Type	Supported
Category	Utilities—Message Information
IA #	2737
Description	This API returns the network header records from a message that originated at a remote site. It gets a list of the message's network header records(similar in format to that produced by LIST^DIC).
Format	QN^XMXUTIL3(xmz[,xmflags][,xmamt][,xmstart],xmtree)
Input Parameters	<p>xmz: (required) Message Internal Entry Number (IEN) in the MESSAGE file (#3.9).</p> <p>xmflags: (optional) Reserved for future use.</p> <p>xmamt: (optional) How many? * (default)—Get all Number—Get this many</p> <p>.xmstart: (optional) Used to start the Lister. The Lister keeps it updated from call to call. ("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.</p> <p>xmtree: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).</p>
Output Parameters	<p>.xmstart: Used to start the Lister. The Lister keeps it updated from call to call. ("IEN") Start <i>after</i> this addressee IEN. Continues from there, with each successive call, to the end. The default is to start at the beginning.</p>

Example

```

>D QN^XMXUTIL3(978437)
>D ^%G

Global ^TMP("XMLIST", $J
      TMP("XMLIST", $J
^TMP("XMLIST", 541073053, 0) = 12^^^0
^TMP("XMLIST", 541073053, 1) = Received: from ISC-SF.VA.GOV by MAILMAN.ISC-
SF.VA.GOV (MailMan/7.1 Turn Around) id 978437 ; 1 May 1998 06:38:29 -0700
(PDT)
^TMP("XMLIST", 541073053, 2) = Received: from FORUM.VA.GOV by ISC-SF.VA.GOV
(MailMan/7.1 TCP/IP-MAILMAN) id 1204177 ; 11 Mar 1998 09:25:25 -0800 (PST)
^TMP("XMLIST", 541073053, 3) = Subject:Released DI*21*43 SEQ #39
^TMP("XMLIST", 541073053, 4) = Date:11 Mar 98 12:22 EST
^TMP("XMLIST", 541073053, 5) = Message-ID:<26463552@FORUM.VA.GOV>
^TMP("XMLIST", 541073053, 6) = From:<National Patch Module@FORUM.VA.GOV>
^TMP("XMLIST", 541073053, 7) = To: PUCE.FIL@FORUM.VA.GOV,
G.PATCH@FORUM.VA.GOV, G.SUPPORT@FORUM.VA.GOV, G.SUPPORT@ISC-ALBANY.VA.GOV,
^TMP("XMLIST", 541073053, 8) = G.SUPPORT@ISC-BIRM.VA.GOV, G.SUPPORT@ISC-
CHICAGO.VA.GOV, G.SUPPORT@ISC-DALLAS.VA.GOV,
^TMP("XMLIST", 541073053, 9) = G.SUPPORT@ISC-SF.VA.GOV, G.SUPPORT@ISC-
SLC.VA.GOV, S.A1AE SERVER VERIFIED@FORUM.VA.GOV,
^TMP("XMLIST", 541073053, 10) = PARKS.RICHARD@FORUM.VA.GOV,
NAPOLI.GERALD@FORUM.VA.GOV, HODGES.BRYAN@FORUM.VA.GOV,
^TMP("XMLIST", 541073053, 11) = G.CSNATHD@FORUM.VA.GOV
^TMP("XMLIST", 541073053, 12) =
Global ^

```

19.2.5 QX^XMXUTIL3(): Local Recipient Extract

Reference Type	Supported
Category	Utilities—Message Information
IA #	2737
Description	This API performs the following actions, depending on the input parameters: Show all users who are current on a message (i.e., read all responses). Show all users who are <i>not</i> current on a message (i.e., have <i>not</i> read all responses). Show all users who have terminated from a message.
Format	QX^XMXUTIL3(xmz,xmflags[,xmamt][,xmstart],xmtrout)
Input Parameters	<p>xmz: (required) This is the</p> <p>xmflags: (required) This flag has three possible values depending on what information you want:</p> <p>C—list users who are current in reading the message.</p> <p>N—list users who are NOT current in reading the message.</p> <p>T" list users who have terminated the message.</p> <p>xmamt: (optional) How many?</p> <p>* (default)—Get all</p> <p>Number—Get this many</p> <p>.xmstart: (optional) Used to start the Lister. The Lister will keep it updated from call to call.</p> <p>("IEN") Start <i>after</i> this line IEN.</p> <p>Continues from there, with each successive call, to the end. The default is to start at the beginning.</p> <p>xmtrout: (required) Target root (closed) to receive the message list. The default is ^TMP("XMLIST", \$J).</p>

Output

.xmstart: Used to start the Lister. The Lister will keep it updated from call to call.
("IEN") Start *after* this line IEN.
Continues from there, with each successive call, to the end. The default is to start at the beginning.

20.0 Glossary

BANNER	A line of text with a user's name and domain that is displayed to everyone who sends mail to the user.
BSCP	Block Mode Simple Communications Protocol, a procedure used for message transmission with error checking.
BULLETIN	A form letter often triggered by a VA FileMan field.
DEVICE	A terminal, printer, modem, or other type of hardware or equipment associated with a computer. A host file of an underlying operating system can be treated like a device in that it can be written to (e.g., for spooling).
DOMAIN	A site for sending and receiving mail.
INITIALIZATION	The process of setting variables in a program to their starting value.
INPUT TRANSFORM	An executable string of M code that is used to check the validity of input and converts it into an internal form for storage.
KEYWORD	A reference name that calls a help frame when entered at a message prompt.
LINE EDITOR	This is VA FileMan's special line-oriented text editor. This editor is used for the word-processing data type.
LOCAL	The system that a user is currently signed on to.
LOG IN/ON	The process of gaining access to a computer system.
LOG OUT/OFF	The process of exiting from a computer system.
MAIL BASKET	Mail baskets provide a way of saving messages in a sorted fashion similar to a filing system. Mail baskets are created at the "Message action" prompt by typing in "S" to save, then the name you wish to call the basket. If the basket already exists, the message will be sent to it. If the basket does not exist, you will be asked if you want it created. Placing a message in a mail basket other than the IN or Waste baskets protects the message from being automatically purged when the IN BASKET PURGE is run.
MESSAGE-ID	A message identifier that shows the time of transmission, the message number and the domain name of the message.
MULTIMEDIA MAIL	Multimedia Mail gives the capability of attaching Binary Large Objects (BLOBs) to electronic messages so that images, spreadsheets, graphs, and other operating system files that are not pure ASCII text, can be sent and received either locally or across the network.
PHYSICAL LINK DEVICE	Hardware used to establish outgoing communication.
"PLAYING A SCRIPT"	A method of opening a transmission link for a message, used to force message transmission of message and testing.
POLLER	An option that opens the transmission line to all domains with "P" in the Flags field.
POSTMASTER	The basket where message queues are stored. Also, the person who manages this basket for a particular site.
PROTOCOL	Code containing logic for opening and closing links, and for sending/receiving transmissions.
PURGE	A procedure used to delete messages or message pointers.

Glossary

QUEUE	A list that stores messages destined for a given domain.
REMOTE	Any system that a user is not signed on to.
SCRIPT	A set of MailMan commands and transmission scripts to a remote domain in the DOMAIN file (#4.2).
SERVER	An automatic mail reader for internal messages.
SMTP	Simple Mail Transport Protocol. The primary transport protocol for MailMan.
SWP	Sliding Window Protocol.
TRANSMISSION SCRIPT	List of commands for directing a message stored in the TRANSMISSION SCRIPT field.
VALIDATION NUMBER	A security check number that must be in the domains of both the sending and receiving sites.
WRAP-AROUND MODE	Text that is fit into available column positions and automatically wraps to the next line, sometimes by splitting at word boundaries (spaces).



REF: For a comprehensive list of commonly used infrastructure- and security-related terms and definitions, please visit the ISS Glossary Web page at the following Web address:

<http://vaww.vista.med.va.gov/iss/glossary.asp>

For a comprehensive list of acronyms, please visit the ISS Acronyms Web site at the following Web address:

<http://vaww.vista.med.va.gov/iss/acronyms/index.asp>

21.0 Appendix A—Message Server Protocol

Overview

A server is an option that is invoked when a mail message that has been addressed to it has been delivered. As an option, many of the parameters associated with the servers are embedded in the definition of the option.



REF: For more information on servers, please refer to "Chapter 12: Servers" in "Part 2: Menu Manager" in the *Kernel Systems Manual*. Options are also described in "Part 2: Menu Manager" in the *Kernel Systems Manual*.

Servers may or may not receive data. The received data usually comes in the form of the text of the message being delivered to it, but the data can also be pointed to by the message and exist in the system, either because it was there in the beginning or because it arrived independently.

Servers can be addressed from a remote site. For example, a server on ALTOONA.MED.VA.GOV can receive a message addressed to it from WASHINGTON.MED.VA.GOV. In fact, this is very common. Because of this capability, an option that has been designated as a server has security features as parameters. Please be aware of these security parameters. Messages addressed to servers will *not* be scheduled, if security is *not* passed.

Filegrams work through the use of a server. Data is loaded into a mail message, addressed and when delivered, processed by the filegram server into the receiving database.

Servers are always invoked through tasks that are set up when the message is delivered into the system, locally or over the network. One of the options is to "Run Immediately." Then the task is scheduled to run "NOW."

However, tasks may not need to be scheduled at all, because the system manager has stated so in the entry for the server in the OPTION file (#19) or because of a problem.

Server Statuses

Server recipients are recorded in the recipient chain of a message and appear similarly to other users. MailMan enters statuses on its own as stages in the server process are reached. First, the message is marked as "Awaiting Server." This indicates that the message has been received and the option is a valid one. At this point, a task has been created to actually invoke MenuMan to schedule or perform the service (option) required.

The last status that MailMan sets is "Served," which means that MenuMan has been called successfully and MenuMan has either performed the task in the case of a server that runs immediately or that some other action has been taken.

At this point, a task could be scheduled to invoke the server, a message could be sent to indicate that the task exists and needs to be scheduled, or some other action that was required was performed. MenuMan has its own statuses that will be used.

\$\$\$SRVTIME^XMS1

This extrinsic function sets the status of recipients in a message.



REF: For details on this API, please refer to Chapter 15, "Servers—Message Activities," in this manual.

Addressing a Server

To address a server, precede the recipient name with "S." (e.g., S.XMECHO, this example sends a message to the MailMan Echo Tester server). "S." must be followed by an option name from the OPTION file (#19) in the Target Domain. If not, a "Recipient not Found" error will occur.

A "Recipient Ambiguous" error will occur, if there is more than one option whose name partially matches the name addressed.

For example, the District Registry server for admitting a new patient could be addressed as follows:

```
S.DGDISTADMIT@SANFRANCISCO.MED.VA.GOV
```

The message is destined for the DGDISTADMIT option at San Francisco. Replies to this message would be from this same name.

Writing a Server Program

The server communicates with mail messages in specific ways. Code is used to interface the server to the message system. The code below returns the original message to the sender:

```
ECHO      ;
K XMY
S XMSUB=$E("Server echo of' "_XMSUB_" `",1,65)
S XMY(XMFROM)=" ",XMTXT="^XMB(3.9,"_XMZ_" ,2,"
D ^XMD
Q
```

In this example, the variable XMFROM contains the sender address and is supplied to the server when invoked. Other variables also exist upon invocation of the server.

Routine ^XMF1 is an example server program supplied with MailMan. ^XMF1 uses some of the other variables supplied to the server.

Execute variable XMREC to read a line of the message. XMER and XMRG are returned.

XMER This variable returns the execution status of XMREC. XMER<0, if there is no more message text to read. The value of XMER will be zero (0), if XMRG is being returned as non-null. XMRG, in that case, will have as its value the text of the next line of the message.

XMRG The value of XMRG will be the next line of message text. XMRG will always be defined, though it will be null when XMER<0.

XMPOS This variable contains the current position of the text returned in the variable XMGR. It is initialized if it is undefined, but should be KILLED by the server when it is finished "Reading" the message.

Here's another example of code, this time from XMF1:

```
S XMA=0
A;
X XMREC ; Receive a line
I $D(XMER) G Q;XMER<0 ; Check for end of message
S XMA=XMA+1 ; Increment local line count
S XMTEXT(XMA)=XMRG ; Set local array
G A ; Go back for another line
```

Double Serving Messages

On occasion, a system backup interrupts the transmission/receive process. It appears to result in the same message being served twice. The Audit Log for the OPTION file (#19) shows two messages with the same message number and subject, but with different Date/Times and Job Numbers.

To avoid this, application servers should be written such that they check for and avoid processing of the same message being delivered to any particular server. MailMan transparently checks this and does *not* deliver twice to mailboxes. However, devices and servers do *not* have mailboxes to check against. Servers can have some understanding of special mail baskets in the Postmaster's mailbox and can be written to check for duplicate deliveries.



REF: For more information on server baskets, please refer to the XMA1C APIs in Chapter 15, "Servers—Message Activities," in this manual.

21.1 Appendix B—Efficient Use of the API

Technical Background

Large messages can be created in a more efficient way using the following method of creating a message.

The simplest and most straightforward approach of creating a message using the API is as follows:

- Load the text of a message into an array
- Set a couple of variables
- D ^XMD

With short messages and if a local array is used, this is fairly efficient. However, when a large message is built, it usually must be stored in a global array. Then, MailMan must read it and rewrite it. This effectively doubles the amount of work the system must do to compile the text of the message. Also, "KILLing" the temporary global array built to store the data passed to the MailMan programmer interface eats up additional resources.

Thus, why not write the text of the message (the data) directly into the message using the available and documented entry points?

Example

The following steps assume that the standard variables already exist in the partition from either Logon or because the job is a TaskMan task.

Step 1—Create a Message with No Text

```
S XMSUB="LARGE DATA TRANSMISSION"; Initialize Subject
S XMDUZ="Application X"           ; Sender
D XMZ^XMA2                       ; Call Create Message Module
I XMZ<1 G RETRY                   ; Abort or retry, if returned value <1
                                ; Make sure you check!
```

Step 2—Put Text into Message

```
S L=0                             ; Initialize Line Counter
A S L=L+1                          ; Increment Line Counter
D data^routine                      ; Create a Line of Data for the Message in 'X'
I $L(X) S ^XMB(3.9,XMZ,2,L,0)=X     ; Put Line of Data into Message
S ^XMB(3.9,XMZ,2,0)="^3.92^"_L_"^"_L_"^"_DT
```

Step 3—Deliver Message to Recipients

```
S XMDUZ="SENDER,LARGEMESSAGE";   A Sender can be free text or you can
                                ;   Leave the variable undefined and the
                                ;   message will appear to be from the
                                ;   user who was logged on.
S XMY("XXX@Q-AUSTIN_'Q'_DOMAIN")="" ; Remote Recipient
S XMY(234567)=""                 ;   Individual as a recipient
.
.
.
D ENT1^XMD ;                     Call for MailMan Delivery
```

The message will now be delivered. This may not happen immediately because the job of delivery the message is passed off to a 'background filer'.

22.0 Appendix C—Looking Up Messages

DIC Lookups

Using DIC lookups into the MESSAGE file (#3.9) is a simple process. The most likely way to do this is by message number (e.g., "123456") or by message subject. In the case where looking the message up by subject is required, the code should understand that message subjects are stored in a case sensitive fashion and the case sensitivity is carried over the network. Putting a "Z" into DIC(0) returns the entire zero node, if it is required. However, the data returned in Y(0) will be raw data and there may be problems with using it directly. It is recommended that calls to the MailMan API be used when extracting data from a field is necessary.

When processing the text of a message, it is recommended that the MailMan Message Server Protocol be used. This means that a message is sent to a serve (S.option_name). When the message is received by the server, which is really a piece of software—an option in the OPTION file (#19), the server protocol can be used. There is also a way to set up and use the executable variable XMREC if needed.



REF: For more information on the MailMan Message Server Protocol, please refer to "Appendix A—Message Server Protocol" in this manual.

Since DIC ensures that only the sender and recipients of a message will be able to find them, the DIC lookup should be used at all times.

23.0 Appendix D—Setting Up Bulletins

What is a Bulletin?

A bulletin consists of a form that can be filled in and an optional mail group. The mail group is normally the first thing that is set up. When code is invoked to send a bulletin, the bulletin will be sent to all the members of that Mail Group. Messages (created from bulletins) can be sent very easily and can be triggered by events in the database, or by code written in a routine by a programmer as in the example of a bulletin cross-reference and an API call (see the examples that follow).

First, in order, set up the entries in the MAIL GROUP file (#3.8) and then in the BULLETIN file (#3.6).

The following is an example of a bulletin:

```
NAME:XMTEST      SUBJECT:TEST BULLETIN TYPE |2|
MESSAGE:  Test Bulletin has been triggered by |1|.
MAIL GROUP:POSTMASTER
DESCRIPTION:  THIS IS A TEST BULLETIN
```

Figure D-1. An Example of a Bulletin

Calling the Programmer API to Send a Bulletin

Variable input into the message text and subject of the message created with a call to the Bulletin API is arranged by the creation of parameters as one enters data into the bulletin fields. Note that there is only one parameter in the message text portion and the subject portion. That parameter is indicated in the MESSAGE field of the BULLETIN file (#3.6) entry by the string "|1|" and the SUBJECT field with the string "|2|". There can be additional parameters. It is only necessary to enter them into the fields. This bulletin could have had much longer text because the MESSAGE field is a word-processing-type field and the variable portion of the text can be on any line of the text, *not* just the first line.

The call to the Bulletin API would then look like this:

```
S XMB="XMTST"
S XMB(1)="XMUSER1,ONE"
S XMB(2)="**User Notification**"
S XMDUZ="ApplicationX"
D ^XMB
```

Figure D-2. Sample code to call to the Bulletin API



REF: Compare this API to the EN^XMB: API in Chapter 10, "Bulletins—Creating and Sending," and the SENDBULL^XMXAPI(): Send a Bulletin and TASKBULL^XMXAPI(): Send a Bulletin APIs in Chapter 4, "Message Actions," in this manual.

Using the example in Figure , this call to the Bulletin API would result in the following:

- A message being sent to the G.POSTMASTER mail group.
- Subject—"TEST BULLETIN TYPE**User Notification**".
- Text—"Test bulletin has been triggered by XMUSER,ONE".



NOTE: The windows (i.e., |1| and |2| in Figure D-1 have been replaced by the values of the corresponding nodes of the XMB Array.

Bulletin Cross-reference

The following dialogue illustrates setting up a Bulletin Cross-reference:

```

Select VA FileMan Option: ENTER or Edit File Entries
INPUT TO WHAT FILE:// BULLETIN
EDIT WHICH FIELD:ALL// <RET>

Select BULLETIN NAME: AFS INSURANCE
Located in the A (Local) namespace.
Are you adding 'AFS INSURANCE' as a new BULLETIN (THE 85TH)? Y <RET> (YES)
SUBJECT: HEALTH INS. FOR PATIENT
Select MAIL GROUP: MCCR DATA COLLECTION
DESCRIPTION:
1>THIS IS A LOCAL BULLETIN. IT NOTIFIES MEMBERS OF THE MCCR DATA
2>COLLECTION MAIL GROUP WHENEVER THE COVERED BY HEALTH INSURANCE
3>QUESTION IN THE PATIENT FILE IS ANSWERED YES.
4><RET>
EDIT Option: <RET>
MESSAGE:
1> PATIENT|1|, SSN#|2|
2>
3>Has had the question, 'COVERED BY HEALTH INSURANCE?' answered 'YES'
4>during the LOAD/EDIT PATIENT DATA process.
5><RET>
EDIT Option: <RET>
Select PARAMETER: 1
DESCRIPTION:
1>PATIENT PATIENT's NAME from File 2
2><RET>
EDIT Option: <RET>
Select PARAMETER: 2
DESCRIPTION:
1>PATIENT's SSN from File 2
2><RET>
EDIT Option: <RET>
Select PARAMETER: <RET>

Select BULLETIN NAME: <RET>

Select VA FileMan Option: UTILIY FUNCTIONS

Select Utility Functions Option: CROSS-Reference a File or Field

MODIFY WHAT FILE:BULLETIN// 2 <RET> PATIENT (27866 Entries)
Select FIELD: COVERED BY HEALTH INSURANCE

NO CURRENT CROSS-REFERENCE
WANT TO CREATE A NEW CROSS-REFERENCE FOR THIS FIELD? NO// Y <RET> (YES)
CROSS-REFERENCE NUMBER:1// <RET>
Select TYPE OF INDEXING:REGULAR// BULLETIN

```

Figure D-3. An Example of Setting Up a Bulletin Cross-reference (1 of 2)

```

---SET LOGIC---

ENTER THE NAME OF A 'BULLETIN' MESSAGE, IF YOU WANT THAT MESSAGE SENT
WHENEVER THE 'COVERED BY HEALTH INSURANCE?' FIELD IS ENTERED OR CHANGED:AFS
INSURANCE
MESSAGE:
1>  PATIENT |1|, SSN#|2|
2>
3>Has had the question, "COVERED BY HEALTH INSURANCE?" answered 'YES'
4>during the LOAD/EDIT PATIENT DATA process.
5><RET>
EDIT Option: <RET>

DO YOU WANT TO MAKE THE SENDING OF 'AFS INSURANCE CONDITIONAL? NO/ Y <RET>
(YES)
ENTER AN EXPRESSION FOR THE CONDITION:COVERED BY HEALTH INSURANCE? ["Y"
<RET> ...OK

ENTER A FIELD NAME (FOR EXAMPLE, 'COVERED BY HEALTH INSURANCE?'), OR A
COMPUTED-FIELD EXPRESSION, THE VALUE OF WHICH WILL BE PASSED INTO THE 'AFS
INSURANCE' MESSAGE, AS PARAMETER #2
  --PATIENT's SSN from File 2
PARAMETER #2: SOCIAL SECURITY NUMBER <RET> ...OK

NOW, IF THE BULLETIN IS TO HAVE 3 OR MORE PARAMETERS INSERTED, ENTER A
FIELD NAME (FOR EXAMPLE, 'COVERED BY HEALTH INSURANCE?'), OR A 'COMPUTED-
FIELD' EXPRESSION, THE VALUE OF WHICH WILL BE PASSED INTO THE 'AFS
INSURANCE' MESSAGE, AS PARAMETER #3
(NOTE THAT NO SUCH PARAMETER IS DEFINED FOR THE 'AFS INSURANCE' BULLETIN)
PARAMETER #3:

---KILL LOGIC---

ENTER THE NAME OF A 'BULLETIN' MESSAGE IF YOU WANT THAT MESSAGE SENT
WHENEVER THE 'COVERED BY HEALTH INSURANCE?' FIELD IS CHANGED OR DELETED:
<RET> NO EFFECT

...CROSS-REFERENCE IS SET

DO YOU WANT TO RUN THE CROSS-REFERENCE FOR EXISTING ENTRIES NOW? NO// N
<RET> (NO)

```

Figure D-4. An Example of Setting Up a Bulletin Cross-reference (2 of 2)

23.1 Index

\$

\$\$ACCESS^XMXSEC API, 3-1
\$\$DM^XMBGRP, 9-1
\$\$EDIT^XMXSEC2 API, 3-1
\$\$HDR^XMGAPI2, 5-2, 19-8
\$\$NU^XM, 8-1
\$\$ORIGIN8R^XMXSEC, 19-11
\$\$SUBCHK^XMGAPI0, 2-9

^

^TMP("XMERR",\$J), 3-1, 3-2, 3-3, 3-4, 4-5, 4-6, 4-7, 4-8, 4-9, 4-10, 4-11, 4-12, 4-13, 4-14, 4-15, 7-2, 7-3, 7-4, 7-6, 7-9, 7-10, 7-11, 9-4, 14-5, 14-6, 14-7, 14-8, 14-9, 14-10, 14-11, 14-12, 14-13, 14-14, 14-15, 14-16, 14-17, 14-18, 14-19, 14-20, 14-21, 14-22
^TMP("XMLIST",\$J), 7-5, 7-8, 19-17, 19-20, 19-21, 19-22
^XMA21 APIs, 11-1
^XMB, 10-1
^XMB APIs, 10-1

A

Acronyms (ISS)
Home Page Web Address, Glossary, 2
Adobe Acrobat Quick Guide Web Address, xix
Adobe Home Page Web Address, xix
APIs
\$\$DM^XMBGRP, 9-1
\$\$EDIT^XMXSEC2, 3-1
\$\$HDR^XMGAPI2, 5-2
\$\$NU^XM, 8-1
\$\$SUBCHK^XMGAPI0, 2-9
^XMA21, 11-1
^XMB, 10-1
INIT^XMVITAE, 13-2
INIT^XMVVITAE, 3-1
INMSG1^XMXUTIL2, 19-9
INMSG2^XMXUTIL2, 3-1
INRESPS^XMXUTIL2, 19-12
KL^XMA1B, 8-4
MAKENEW^XMXUTIL, 17-3
OPTEDIT^XMXSEC2, 3-1
OPTMSG^XMXSEC2, 3-1

QD^XMXUTIL3, 19-17
QL^XMXUTIL3, 19-20
REC^XMS3, 5-7
XMZ^XMA2, 2-3

Arrays
XMV, 3-1

B

BULLETIN File (#3.6), 10-1, 10-2, 10-3
D, 1

C

Callout Boxes, xvi
COMMUNICATIONS PROTOCOL File (#3.4), 5-5

D

Data Dictionary
Data Dictionary Utilities Menu, xviii
DIFROM Variable, 2-5, 2-6, 2-11, 4-3
Documentation
Symbols, xv
DOMAIN File (#4.2), 19-19
Glossary, 2

E

EVS Anonymous Directories, xix

F

Fields
FORWARDED BY (#8)
MESSAGE File (#3.9), 4-2
FROM (#1)
MESSAGE File (#3.9), 2-3, 4-2
INFORMATION ONLY? (#1.97)
MESSAGE File (#3.9), 2-3
LAST READ DATE/TIME (#2)
MESSAGE File (#3.9), 15-2
MESSAGE, 1
RECIPIENT Multiple
MESSAGE File (#3.9), 4-2, 15-2
ROUTINE, 13-2
SENDER (#1)
MESSAGE File (#3.9), 4-2

STATUS (#5)
MESSAGE File (#3.9), 15-2
SUBJECT (#.01)
MESSAGE File (#3.9), 2-3
Files
BULLETIN (#3.6), 10-1, 10-2, 10-3
D, 1
COMMUNICATIONS PROTOCOL (#3.4),
5-5
DOMAIN (#4.2), 19-19
Glossary, 2
MAIL GROUP (#3.8), 9-1, 9-3, 9-4, 9-5
D, 1
MAILBOX (#3.7), 17-4
MESSAGE (#3.9), 1-3, 2-1, 2-2, 2-3, 2-5, 2-6,
2-7, 2-8, 2-9, 2-11, 3-1, 3-2, 3-3, 3-4, 4-2,
4-4, 4-5, 4-6, 4-7, 4-11, 4-12, 4-13, 4-15, 5-
1, 5-4, 5-6, 5-7, 6-1, 6-2, 6-3, 6-4, 7-6, 8-3,
8-4, 8-5, 8-6, 10-3, 13-4, 14-5, 14-6, 14-7,
14-8, 14-9, 14-10, 14-11, 14-12, 14-13, 14-
14, 14-15, 14-17, 14-18, 14-20, 14-22, 15-
1, 15-2, 15-3, 16-2, 16-3, 17-2, 17-3, 17-5,
19-6, 19-7, 19-9, 19-11, 19-12, 19-13, 19-
14, 19-15, 19-16, 19-17, 19-20, 19-21
C, 1
NEW PERSON (#200), 17-3
OPTION (#19), 1
A, 2, 3
FIND^DIC, 19-17, 19-20
FORWARDED BY Field (#8)
MESSAGE File (#3.9), 4-2
FROM Field (#1)
MESSAGE File (#3.9), 2-3, 4-2
FTP, xix

G

Glossary
ISS Home Page Web Address, Glossary, 2

H

Help
Question Marks, xvii
Home Pages
Adobe Acrobat Quick Guide Web Address,
xix
Adobe Web Address, xix
HSD&D Home Page Web Address, xviii
ISS

Acronyms Home Page Web Address,
Glossary, 2
Glossary Home Page Web Address,
Glossary, 2
MailMan Home Page Web Address, xix
VistA Documentation Library (VDL) Home
Page Web Address, xix
HSD&D
Home Page Web Address, xviii

I

INFORMATION ONLY? Field (#1.97)
MESSAGE File (#3.9), 2-3
INIT^XMVITAE API, 13-2
INIT^XMVVITAE API, 3-1
INMSG^XMXUTIL2, 19-8
INMSG1^XMXUTIL2, 14-22, 17-2, 17-3, 19-7,
19-9, 19-12
INMSG1^XMXUTIL2 API, 19-7, 19-8
INMSG2^XMXUTIL2, 14-22, 17-3
INMSG2^XMXUTIL2 API, 3-1, 19-7, 19-8
INRESPTS^XMXUTIL2, 17-3, 19-12
ISS
Acronyms
Home Page Web Address, Glossary, 2
Glossary
Home Page Web Address, Glossary, 2

K

KL^XMA1B, 8-4

L

LAST NEW MSG NOTIFY DATE/TIME field,
17-4
LAST READ DATE/TIME Field (#2)
MESSAGE File (#3.9), 15-2
List File Attributes Option, xviii
LIST^DIC, 19-16, 19-17, 19-20, 19-21
LIST^DIC API, 7-6
Lister, 7-4, 7-7, 19-16, 19-17, 19-20, 19-21, 19-
22, 19-23

M

MAIL GROUP File (#3.8), 9-1, 9-3, 9-4, 9-5
D, 1
MAILBOX File (#3.7), 17-4
MailMan Home Page Web Address, xix
MAKENEW^XMXUTIL, 17-3

Menus

Data Dictionary Utilities, xviii

MESSAGE Field, 1

MESSAGE File (#3.9), 1-3, 2-1, 2-2, 2-3, 2-5,
2-6, 2-7, 2-8, 2-9, 2-11, 3-1, 3-2, 3-3, 3-4, 4-2,
4-4, 4-5, 4-6, 4-7, 4-11, 4-12, 4-13, 4-15, 5-1,
5-4, 5-6, 5-7, 6-1, 6-2, 6-3, 6-4, 7-6, 8-3, 8-4,
8-5, 8-6, 10-3, 13-4, 14-5, 14-6, 14-7, 14-8,
14-9, 14-10, 14-11, 14-12, 14-13, 14-14, 14-
15, 14-17, 14-18, 14-20, 14-22, 15-1, 15-2,
15-3, 16-2, 16-3, 17-2, 17-3, 17-5, 19-6, 19-7,
19-9, 19-11, 19-12, 19-13, 19-14, 19-15, 19-
16, 19-17, 19-20, 19-21

C, 1

Modems, 1

N

Network Signature, 4-7

NEW PERSON File (#200), 17-3

O

OPTEDIT^XMXSEC2 API, 3-1

OPTION File (#19), 1

A, 2, 3

Options

Data Dictionary Utilities, xviii

List File Attributes, xviii

Read/Manage Messages, 8-2

Send a Message, 2-7

XMPOST, 10-2

XMREAD, 8-2

XMSEND, 2-1, 2-7

OPTMSG^XMXSEC2 API, 3-1

Q

QD^XMXUTIL3, 19-17

QL^XMXUTIL3, 19-20

QMBOX^XMXAPI, 17-4

Question Mark Help, xvii

R

Read/Manage Messages Option, 8-2

REC^XMS3, 5-7

RECIPIENT Multiple Field

MESSAGE File (#3.9), 4-2, 15-2

Reference Type

Supported

\$\$DM^XMBGRP, 9-1

\$\$HDR^XMGAPI2, 5-2

\$\$NU^XM, 8-1

\$\$SUBCHK^XMGAPI0, 2-9

^XMB, 10-1

INMSG1^XMXUTIL2, 19-9

INRESPS^XMXUTIL2, 19-12

KL^XMA1B, 8-4

MAKENEW^XMXUTIL, 17-3

QD^XMXUTIL3, 19-17

QL^XMXUTIL3, 19-20

REC^XMS3, 5-7

XMZ^XMA2, 2-3

ROUTINE Field, 13-2

S

Security Keys

XM GROUP PRIORITY, 14-19

XMMGR, 14-17

Send a Message Option, 2-7

SENDER Field (#1)

MESSAGE File (#3.9), 4-2

SHARED,MAIL, 4-3, 14-10, 14-19

STATUS Field (#5)

MESSAGE File (#3.9), 15-2

SUBJECT Field (#.01)

MESSAGE File (#3.9), 2-3

Symbols

Found in the Documentation, xv

U

URLs

Adobe Acrobat Quick Guide Web Address,
xix

Adobe Home Page Web Address, xix

HSD&D Home Page Web Address, xviii

ISS

Acronyms Home Page Web Address,

Glossary, 2

Glossary Home Page Web Address,

Glossary, 2

MailMan Home Page Web Address, xix

VistA Documentation Library (VDL) Home

Page Web Address, xix

V

Variables

DIFROM, 2-5, 2-6, 2-11, 4-3

VistA Documentation Library (VDL)

Home Page Web Address, xix

W

Web Pages

- Adobe Acrobat Quick Guide Web Address, xix
- Adobe Home Page Web Address, xix
- HSD&D Home Page Web Address, xviii
- ISS
 - Acronyms Home Page Web Address, Glossary, 2
 - Glossary Home Page Web Address, Glossary, 2
- MailMan Home Page Web Address, xix
- VistA Documentation Library (VDL) Home Page Web Address, xix

X

XM

- \$\$NU^XM, 8-1
- XM GROUP PRIORITY Security Key, 14-19
- XMA1B
 - KL^XMA1B, 8-4
- XMA2
 - XMZ^XMA2, 2-3
- XMB
 - ^XMB, 10-1
- XMBGRP

- \$\$DM^XMBGRP, 9-1
- XMERR, 3-1, 3-2, 3-3, 3-4, 4-5, 4-6, 4-7, 4-8, 4-9, 4-10, 4-11, 4-12, 4-13, 4-14, 4-15, 7-2, 7-3, 7-4, 7-6, 7-9, 7-10, 7-11, 9-4, 14-5, 14-6, 14-7, 14-8, 14-9, 14-10, 14-11, 14-12, 14-13, 14-14, 14-15, 14-16, 14-17, 14-18, 14-19, 14-20, 14-21, 14-22
- XMGAPI0
 - \$\$SUBCHK^XMGAPI0, 2-9
- XMGAPI2
 - \$\$HDR^XMGAPI2, 5-2
- XMINSTR, 3-1
- XMMGR Security Key, 14-17
- XMPOST option, 10-2
- XMREAD Option, 8-2
- XMS3
 - REC^XMS3, 5-7
- XMSEND option, 2-1, 2-7
- XMV Array, 3-1
- XXUTIL
 - MAKENEW^XXUTIL, 17-3
- XXUTIL2
 - INMSG1^XXUTIL2, 19-9
 - INRESPTS^XXUTIL2, 19-12
- XXUTIL3
 - QD^XXUTIL3, 19-17
 - QL^XXUTIL3, 19-20
- XMZ^XMA2, 2-3