EMAIL$ - Send Electronic Mail

The EMAIL$ routine allows you to send electronic mail to one, or more, recipients.

1. Invocation

To send an e-mail using EMAIL$ code:

```
CALL EMAIL$ USING em flag
```

where *flag* is a PIC 9(4) COMP field or literal as described below; *em* is the Email Control Block defined as follows:

```
01 EM
  02      EMSUB   OCCURS 10                * Up to 10 recipients
    03    EMRLEN           PIC 9(4) COMP   * Length of each recipient
    03    EMRPTR           PIC PTR         * Pointer to recipient text
   02     EMSLEN           PIC 9(4) COMP   * Length of subject text
   02     EMSPTR           PIC PTR         * Pointer to subject text
   02     EMTLEN           PIC 9(4) COMP   * Length of text block
   02     EMTPTR           PIC PTR         * Pointer to text block
   02     EMATT   OCCURS 20                * List of up to 20 attachments
    03    EMALEN           PIC 9(4) COMP   * Length of attachment
    03    EMAPTR           PIC PTR         * Pointer to attachment pathname
   02     EMCCS   OCCURS 10                * List of up to 10 CC's
    03    EMCLEN           PIC 9(4) COMP   * Length of CC recipient
    03    EMCPTR           PIC PTR         * Pointer to CC recipient text
   02     EMBCCS  OCCURS 10                * List of up to 10 BCC's
    03    EMBLEN           PIC 9(4) COMP   * Length of BCC recipient
    03    EMBPTR           PIC PTR         * Pointer to BCC recipient text
```

For GX V4.4q, and earlier, the *flag* variable is a simple Boolean flag:

0	the e-mail dialogue box is to always be displayed;
1	the e-mail dialogue box is only to displayed if there are no recipients.

For GX V4.4r, and later, the *flag* variable is a collection of bit-masks:

#01	0	the e-mail dialogue box is to always be displayed
#01	1	the e-mail dialogue box is only to displayed if there are no recipients
#02	0	no Read Receipt required
#02	1	Read Receipt required
#04		Reserved for future use
#08		Reserved for future use
#10		Reserved for future use
#20		Reserved for future use
#40		Reserved for future use
#80		Reserved for future use

The code in GX.EXE and GSMWIN32.EXE that sends the email uses the MAPI interface. To check for the presence of the MAPI interface on the GX (GSMWIN32) PC code:

```
CALL EMAIL$
```

2. STOP Codes and Exception Conditions

The following STOP codes may be generated by EMAIL$:

STOP code	Description
13602	EMAIL$ has been called by an application that is not running on GX.EXE or GSMWIN32.EXE.
13608	The total length of the GX command block has exceeded 8K.

The following EXIT codes may be returned by EMAIL$:

EXIT code	$$COND	Description
13604	4	Unable to allocate memory for temporary work buffer
13606	6	The MAPI interface is not available on the PC running GX.EXE or GSMWIN32.EXE.
13607	7	GX has returned an error when sending the email. The error code from Simple MAPI is returned in $$CRES: 0 SUCCESS_SUCCESS 1 MAPI_USER_ABORT 2 MAPI_E_FAILURE 3 MAPI_E_LOGIN_FAILURE 4 MAPI_E_DISK_FULL 5 MAPI_E_INSUFFICIENT_MEMORY 6 MAPI_E_ACCESS_DENIED 7 RESERVED 8 MAPI_E_TOO_MANY_SESSIONS 9 MAPI_E_TOO_MANY_FILES 10 MAPI_E_TOO_MANY_RECIPIENTS 11 MAPI_E_ATTACHMENT_NOT_FOUND 12 MAPI_E_ATTACHMENT_OPEN_FAILURE 13 MAPI_E_ATTACHMENT_WRITE_FAILURE 14 MAPI_E_UNKNOWN_RECIPIENT 15 MAPI_E_BAD_RECIPTYPE 16 MAPI_E_NO_MESSAGES 17 MAPI_E_INVALID_MESSAGE 18 MAPI_E_TEXT_TOO_LARGE 19 MAPI_E_INVALID_SESSION 20 MAPI_E_TYPE_NOT_SUPPORTED 21 MAPI_E_AMBIGUOUS_RECIPIENT 22 MAPI_E_MESSAGE_IN_USE 23 MAPI_E_NETWORK_FAILURE 24 MAPI_E_INVALID_EDITFIELDS 25 MAPI_E_INVALID_RECIPS 26 MAPI_E_NOT_SUPPORTED For a list of error codes from the Extended MAPI interface see below.

3. Programming Notes

EMAIL$ is only available when running on GX.EXE or GSMWIN32.EXE. Any attempt to use EMAIL$ on any other terminal will result in a STOP code.

EMAIL$ uses the Simple MAPI interface. An email client which supports Simple MAPI **MUST** be installed on the GX PC. Note that the Extended MAPI interface is not currently supported.

For GX V4.1m, and later, the %XPPDF string can be used to specify the folder used by GX.EXE when creating a PDF file using the Extended PF Printing option (see Technical Note IN343). **Important Note:** The Extended PF Print operation and the subsequent call to EMAIL$ must be performed in the same partition.

As explained above, when EMAIL$ is used in conjunction with GX V4.4r, or later, it is possible to request a Read Receipt. The return of a Read Receipt will depend on the email settings of recipients email client. Note that is it not possible to request a Delivery Receipt programmatically.

GSM SP-30 Intercept Routine

For GSM SP-30, and later, EMAIL$ supports an Intercept Routine that can be used by application code to record and log, **BUT NOT ALTER,** EMAIL$ activity. To enable the Intercept Routine, initialise the pointer $$EMINT to the address of the routine. For example:

```
POINT $$EMINT AT EMINTERCEPT
```

The user-written Intercept Routine MUST be coded as follows:

```
SECTION EMINTERCEPT
$SET "EMAIL$ INTERCEPT"
ENTRY USING L-P1
ON NO EXCEPTION
*
* Process single-parameter call (see below)
*
ELSE
ENTRY USING L-P2A L-P2B
*
* Process 2-parameter call (see below)
*
END
EXIT
```

The slightly unconventional entry-point code is to allow EMAIL$ to perform validation on the $$EMINT pointer before passing control to the Intercept Routine.

The Intercept Routine is called with a single PIC 9(4) COMP parameter (e.g. L-P1) just before EMAIL$ exits with either an EXIT statement or an EXIT with *N* exception. The value of the passed parameter will be one of:

Value	Description

0	Normal exit from EMAIL$. No error condition detected.
4	EMAIL$ was unable to allocate memory for temporary work buffer.
6	The MAPI interface is not available on the PC running GX.EXE.
7	GX has returned an error when sending the email. The error code from Simple MAPI is returned in $$CRES (see above for possible $$CRES values).

The Intercept Routine is called with two parameters immediately after the EMAIL$ entry point. The two parameters are the *em* Email Control Block (L-P2A); and the *flag*, a PIC 9(4) COMP field or literal (L-P2B). Both of these parameters are fully described at the start of this document.

In all cases the Intercept Routine must never attempt to alter any of the passed parameters and must always exit WITHOUT an exception.

The Intercept Routine pointer, $$EMINT, must be reset to HIGH-VALUES if no further intercepts are required. **$$EMINT must also be reset to HIGH-VALUES in the UNLOAD DIVISION of any frame that has initialised the pointer.**

Pre-requisites

Although other factors (e.g. invalid EM control block, illegal email address, too many attachments etc.) may prevent a successfully email transmission three basic **minimum** requirement must be satisfied:

- A Simple-MAPI compliant e-mail service (e.g. Microsoft Outlook) must be configured on the PC running GX (the supported versions of Microsoft Outlook are described below);
- The MAPI DLL (normally mapi32.dll) must be installed and registered on the PC running GX;
- The "MAPIInterface" setting in the [email] section of the GX.INI file be set to "On". This setting specifies whether the MAPI interface should be enabled in GX. Note that this setting should only be enabled if **both** the other 2 requirements are met.

MAPI client available	**Mapi32.dll installed**	**GX.INI MAPIInterface setting**	**Results from EMAIL$**
No	No	No	CALL EMAIL$ will return an exception
No	Yes	No	CALL EMAIL$ will return an exception
Yes	No	No	CALL EMAIL$ will return an exception
Yes	Yes	No	CALL EMAIL$ will return an exception
No	No	Yes	GX will display a warning message at start-up; CALL EMAIL$ will return an exception
No	Yes	Yes	No warning message from GX; CALL EMAIL$ will succeed; CALL EMAIL$ USING EM will always

			return an exception. The mapi32.dll will display a warning pop-up.
Yes	No	Yes	GX will display a warning message at start-up; CALL EMAIL$ will return an exception
Yes	Yes	Yes	CALL EMAIL$ will succeed; CALL EMAIL USING EM will succeed (subject to a valid email address etc.)

Extended Error Codes

In addition to the result codes returned in $$CRES listed above the following values may also be returned:

Error condition	$$CRES hex value	$$CRES decimal value (GSM SP-31, or lower)	$$CRES decimal value (GSM SP-32, or higher)
MAPI_W_ERRORS_RETURNED	#00040380	263040	263040
MAPI_W_POSITION_CHANGED	#00040481	263297	263297
MAPI_W_APPROX_COUNT	#00040482	263298	263298
MAPI_W_CANCEL_MESSAGE	#00040580	263552	263552
MAPI_W_PARTIAL_COMPLETION	#00040680	263808	263808
MAPI_E_INTERFACE_NOT_SUPPORTED	#80004002	Overflow	16386
MAPI_E_CALL_FAILED	#80004005	Overflow	16389
MAPI_E_NO_SUPPORT	#80040102	Overflow	262402
MAPI_E_BAD_CHARWIDTH	#80040103	Overflow	262403
MAPI_E_STRING_TOO_LONG	#80040105	Overflow	262405
MAPI_E_UNKNOWN_FLAGS	#80040106	Overflow	262406
MAPI_E_INVALID_ENTRYID	#80040107	Overflow	262407
MAPI_E_INVALID_OBJECT	#80040108	Overflow	262408
MAPI_E_OBJECT_CHANGED	#80040109	Overflow	262409
MAPI_E_OBJECT_DELETED	#8004010A	Overflow	262410
MAPI_E_BUSY	#8004010B	Overflow	262411
MAPI_E_NOT_ENOUGH_DISK	#8004010D	Overflow	262413
MAPI_E_NOT_ENOUGH_RESOURCES	#8004010E	Overflow	262414
MAPI_E_NOT_FOUND	#8004010F	Overflow	262415
MAPI_E_VERSION	#80040110	Overflow	262416
MAPI_E_LOGON_FAILED	#80040111	Overflow	262417
MAPI_E_SESSION_LIMIT	#80040112	Overflow	262418
MAPI_E_USER_CANCEL	#80040113	Overflow	262419
MAPI_E_UNABLE_TO_ABORT	#80040114	Overflow	262420
MAPI_E_NETWORK_ERROR	#80040115	Overflow	262421
MAPI_E_DISK_ERROR	#80040116	Overflow	262422
MAPI_E_TOO_COMPLEX	#80040117	Overflow	262423
MAPI_E_BAD_COLUMN	#80040118	Overflow	262424
MAPI_E_EXTENDED_ERROR	#80040119	Overflow	262425
MAPI_E_COMPUTED	#8004011A	Overflow	262426
MAPI_E_CORRUPT_DATA	#8004011B	Overflow	262427
MAPI_E_UNCONFIGURED	#8004011C	Overflow	262428
MAPI_E_FAILONEPROVIDER	#8004011D	Overflow	262429
MAPI_E_UNKNOWN_CPID	#8004011E	Overflow	262430
MAPI_E_UNKNOWN_LCID	#8004011F	Overflow	262431
MAPI_E_PASSWORD_CHANGE_REQUIRED	#80040120	Overflow	262432
MAPI_E_PASSWORD_EXPIRED	#80040121	Overflow	262433
MAPI_E_INVALID_WORKSTATION_ACCOUNT	#80040122	Overflow	262434
MAPI_E_INVALID_ACCESS_TIME	#80040123	Overflow	262435
MAPI_E_ACCOUNT_DISABLED	#80040124	Overflow	262436
MAPI_E_END_OF_SESSION	#80040200	Overflow	262656
MAPI_E_UNKNOWN_ENTRYID	#80040201	Overflow	262657
MAPI_E_MISSING_REQUIRED_COLUMN	#80040202	Overflow	262658
MAPI_E_BAD_VALUE	#80040301	Overflow	262913
MAPI_E_INVALID_TYPE	#80040302	Overflow	262914
MAPI_E_TYPE_NO_SUPPORT	#80040303	Overflow	262915
MAPI_E_UNEXPECTED_TYPE	#80040304	Overflow	262916

MAPI_E_TOO_BIG	#80040305	Overflow	262917
MAPI_E_DECLINE_COPY	#80040306	Overflow	262918
MAPI_E_UNEXPECTED_ID	#80040307	Overflow	262919
MAPI_E_UNABLE_TO_COMPLETE	#80040400	Overflow	263168
MAPI_E_TIMEOUT	#80040401	Overflow	263169
MAPI_E_TABLE_EMPTY	#80040402	Overflow	263170
MAPI_E_TABLE_TOO_BIG	#80040403	Overflow	263171
MAPI_E_INVALID_BOOKMARK	#80040405	Overflow	263173
MAPI_E_WAIT	#80040500	Overflow	263424
MAPI_E_CANCEL	#80040501	Overflow	263425
MAPI_E_NOT_ME	#80040502	Overflow	263426
MAPI_E_CORRUPT_STORE	#80040600	Overflow	263680
MAPI_E_NOT_IN_QUEUE	#80040601	Overflow	263681
MAPI_E_NO_SUPPRESS	#80040602	Overflow	263682
MAPI_E_COLLISION	#80040604	Overflow	263684
MAPI_E_NOT_INITIALIZED	#80040605	Overflow	263685
MAPI_E_NON_STANDARD	#80040606	Overflow	263686
MAPI_E_NO_RECIPIENTS	#80040607	Overflow	263687
MAPI_E_SUBMITTED	#80040608	Overflow	263688
MAPI_E_HAS_FOLDERS	#80040609	Overflow	263689
MAPI_E_HAS_MESSAGES	#8004060A	Overflow	263690
MAPI_E_FOLDER_CYCLE	#8004060B	Overflow	263691
MAPI_E_STORE_FULL	#8004060C	Overflow	263692
MAPI_E_AMBIGUOUS_RECIP	#80040700	Overflow	263936
MAPI_E_INVALID_PARAMETER	#80070057	Overflow	458839
MAPI_E_NO_ACCESS	#80070005	Overflow	458757
MAPI_E_NOT_ENOUGH_MEMORY	#8007000E	Overflow	458766

Supported Versions of Microsoft Outlook

EMAIL$ uses the Simple MAPI interface.

Settings in the GX.INI file [email] section

EMAIL$ uses the Simple MAPI interface.

4. Examples

[EXAMPLE REQUIRED]

5. Copy-Books

See copy-book "$E" in copy-library S.SYS32. Note that this copy-book **MUST** be expanded using a SUBSTITUTING clause. For example:

```
COPY "$E" USING "EM"
```

6. See Also

None.