

Msg Function

Description

Displays dynamically constructed system messages.

Syntax

*variable*return = **Msg**(ownerwindow, msgstructure, msgname, reserved, parameters)

Parameters

The Msg function has the following parameters.

Parameter	Description
Ownerwindow	Msg requires a parent window. Note: When using an MDIFrame, the ownerwindow will always be the frame regardless of the value passed to the variable. To display a message without being in event context (for example, if code is executed from the system editor) use the Windows API function MessageBox. This requires SYSTEM MODAL to be set and all string parameters to be terminated with char(0). An older version of Msg did not require a parent window. This older version is still shipped with OpenInsight in order to support existing software but has been renamed as Msg2. Msg2() is automatically called if the message is displayed outside of event context.
Msgstructure	Contains the message definition. This structure is detailed table below. If msgname is passed, the message details that are passed in msgstructure are used to override the corresponding parts of the message definition stored in the repository.
Msgname	To display a message stored in the repository, pass either the name of the message or the message's entity id.
Reserved	This value is reserved; pass "".
Parameters	An @fm-delimited list of substitutable parameter values. Within the message text, the string "%1%" is replaced with the first substitutable parameter, "%2%" is replaced with the second, and so forth.

Remarks

Note: The Msg function resets the Set_Status() code to zero (0).

Use Msg either for display messages (i.e. error handling) or use it for gathering very small scale user input, as for example with Continue? Yes No situations.

The return value depends on the value passed for the type field of the msgstructure argument.

Note: If you are trying to use the MSG function and the variable you are trying to display contains any system delimiters it may not display correctly. (for example, a message box or button may display with garbled text or won't function properly) What you need to do is use either the [CONVERT](#) or [SWAP](#) statements.

For example:

```
CONVERT @fm to ', ' in VarName
```

-or-

```
SWAP @fm with ', ' in VarName
```

Your message box should now display correctly.

Message Structure

Use the [MSG_EQUATES](#) insert record for access to the values in the following table.

Field Position	Name	Purpose
1	text	" (the vertical bar) or @tm. Default is null.

2

type

Message types:
Information Types

Types	Buttons Displayed on Message
BO	OK
BOC	OK/Cancel
BNY	Yes/No
BNYC	Yes/No/Cancel
BRC	Retry/Candle
BAR	Abort/Retry
BARI	Abort/Retry/Ignore
B{LIST}	User defined buttons, where {LIST} is a comma-delimited list of button text, e.g. "BStop,Continue"

Response Types

Types	Description
R	Response message, returning default input if escape pressed.
RC	Same as R, but converting response to upper case.
RCE	Same as RE, but converting response to upper case.
RE	Response message, returning Char(27) if escape pressed.

Timed Types

Types	Description
T{n}	Timed message of n seconds, eg. "T2.5".
TA{n}	Asynchronous timed message (splash screen) of n seconds, eg. "TA10". Note: The message will not close until the engine is idle unless the Yield() function is called.

Processing Types

Types	Description
U	Displays the message and returns immediately (see example below).
D	Closes a processing message; the return value from a U-type message is a D-type message structure which will close the message.

Miscellaneous Types

Types	Description
N	Prefixing any of the message types with "N" suppresses the message display, returning the message's default value. This can be used to change messages from interactive to non-interactive for batch processes; for example, instead of "BARI", pass "NBARI" (meaning don't display the abort/retry/ignore message)

3

modality

Value	Description
W	Window. The message's parent window is disabled while the message is displayed.
A	Application. The OpenInsight application is disabled while the message is displayed.
S	System. All windows from all applications are disabled while the message is displayed.

4	icon	<div>The image to display on the Msg.</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>Null</td><td>None</td></tr><tr><td>*</td><td>Asterisk (Info)</td></tr><tr><td>?</td><td>Question</td></tr><tr><td>!</td><td>Warning</td></tr><tr><td>H</td><td>Halt</td></tr><tr><td>B</td><td>User-specified bitmap (specified in the bitmap field)</td></tr></table>	Value	Description	Null	None	*	Asterisk (Info)	?	Question	!	Warning	H	Halt	B	User-specified bitmap (specified in the bitmap field)
Value	Description															
Null	None															
*	Asterisk (Info)															
?	Question															
!	Warning															
H	Halt															
B	User-specified bitmap (specified in the bitmap field)															
5	default button	1, 2, or 3 depending on number of actual buttons. Response messages don't have a default button; the edit control has the focus. Default is 1.														
6	col	x position to place the dialog box. A value of -1 centers within the message's parent window. A value of -2 centers with respect to the screen. If either col or row is -1 or -2, then the message is centered accordingly, both horizontally and vertically.														
7	row	y position to place the dialog box. A value of -1 centers within the message's parent window. A value of -2 centers with respect to the screen. If either col or row is -1 or -2, then the message is centered accordingly, both horizontally and vertically.														
8	justification	<div>Message text justification.</div> <table><tr><th>Justification</th><th>Description</th></tr><tr><td>T</td><td>Text (default). Text Formatting with word wrap.</td></tr><tr><td>L</td><td>Left</td></tr><tr><td>R</td><td>Right</td></tr><tr><td>C</td><td>Center</td></tr></table>	Justification	Description	T	Text (default). Text Formatting with word wrap.	L	Left	R	Right	C	Center				
Justification	Description															
T	Text (default). Text Formatting with word wrap.															
L	Left															
R	Right															
C	Center															
9	bkcolor	<div>@VM-delimited list specifying R,G,B background color. For example: "255:@VM:255:@VM:255" is White, "255:@VM:0:@VM:0" is Red.</div> <div>Default is white.</div>														
10	fgcolor	@VM-delimited list specifying R,G,B foreground color. Default, "0:@VM:0:@VM:255", is blue.														
11	text width	The message width; for response messages, this specifies the response field width.														
12	caption	Dialog box caption text. Default is "Message."														
13	validation	Validation pattern to check response message return values against with IN.VALUE. Default is null.														
14	default input	Default input to be put into edit control in a response message. (Default is null.)														
15	mask input	<div>A boolean value used to determine if the input should be masked with asterisks? This is used for password entry.</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>No masking</td></tr><tr><td>1</td><td>Masked</td></tr></table>	Value	Description	0	No masking	1	Masked								
Value	Description															
0	No masking															
1	Masked															
16	bitmap	Name of a bitmap registered in the Repository. Used when icon is set to "B".														
17	clip bitmap	Boolean value, true to clip bitmaps, false to resize; see the IMAGECLIP property for more detail.														
18	font	Font structure for the text of the message; see the FONT property for more detail.														

19	literal function	<p>Boolean value. When TRUE the Msg() function assumes that the default value is the name of a function which returns the default value for the message.</p> <p>For example, if you wrote a function called CURRENTUSER which returned the user name of the current user, you could specify CURRENTUSER as the default value (MDEFINPUT\$) and set MLITERAL\$ to true, so that the current user name would be the default value for the message; parameters are passed to the specified function depending on the number of parameters that are supported by the function</p> <table><tr><th># of Params</th><th>Values Passed in Parameter</th></tr><tr><td>0</td><td>None</td></tr><tr><td>1</td><td>MsgName</td></tr><tr><td>2+</td><td>MsgName, MsgStructure</td></tr></table>	# of Params	Values Passed in Parameter	0	None	1	MsgName	2+	MsgName, MsgStructure															
# of Params	Values Passed in Parameter																								
0	None																								
1	MsgName																								
2+	MsgName, MsgStructure																								
20	help	<p>Used to add a Help button the message. The structure of the field is an @vm-delimited array of Help information.</p> <table><tr><th>Position</th><th>Description</th></tr><tr><td><20,1></td><td><p>Help Type</p><table><tr><th>Type</th><th>Description</th><th>Specifier</th></tr><tr><td>Q</td><td>QuickHelp</td><td>Name of the APPNOTE entity.</td></tr><tr><td>M</td><td>Message</td><td>Name of the MSG entity</td></tr><tr><td>H</td><td>WinHelp</td><td>HelpFile, HelpId</td></tr><tr><td>S</td><td>Stored Procedure</td><td>Procedure Name[, param1..]</td></tr></table></td></tr><tr><td><20,2></td><td>Help specifier that is specific to the above Help Types.</td></tr><tr><td><20,3></td><td><p>Button text. Defaults to "&Help".</p><p>Note: Specify the AppNote, Message, HelpFile, or ProcName as it appears in the repository outliner. For example, the OINSIGHT.HLP file is registered as OINSIGHT, so specify the HelpFile as "OINSIGHT" (look in the outline under "General", "Windows Components", "Help Files")</p></td></tr></table>	Position	Description	<20,1>	<p>Help Type</p> <table><tr><th>Type</th><th>Description</th><th>Specifier</th></tr><tr><td>Q</td><td>QuickHelp</td><td>Name of the APPNOTE entity.</td></tr><tr><td>M</td><td>Message</td><td>Name of the MSG entity</td></tr><tr><td>H</td><td>WinHelp</td><td>HelpFile, HelpId</td></tr><tr><td>S</td><td>Stored Procedure</td><td>Procedure Name[, param1..]</td></tr></table>	Type	Description	Specifier	Q	QuickHelp	Name of the APPNOTE entity.	M	Message	Name of the MSG entity	H	WinHelp	HelpFile, HelpId	S	Stored Procedure	Procedure Name[, param1..]	<20,2>	Help specifier that is specific to the above Help Types.	<20,3>	<p>Button text. Defaults to "&Help".</p> <p>Note: Specify the AppNote, Message, HelpFile, or ProcName as it appears in the repository outliner. For example, the OINSIGHT.HLP file is registered as OINSIGHT, so specify the HelpFile as "OINSIGHT" (look in the outline under "General", "Windows Components", "Help Files")</p>
Position	Description																								
<20,1>	<p>Help Type</p> <table><tr><th>Type</th><th>Description</th><th>Specifier</th></tr><tr><td>Q</td><td>QuickHelp</td><td>Name of the APPNOTE entity.</td></tr><tr><td>M</td><td>Message</td><td>Name of the MSG entity</td></tr><tr><td>H</td><td>WinHelp</td><td>HelpFile, HelpId</td></tr><tr><td>S</td><td>Stored Procedure</td><td>Procedure Name[, param1..]</td></tr></table>	Type	Description	Specifier	Q	QuickHelp	Name of the APPNOTE entity.	M	Message	Name of the MSG entity	H	WinHelp	HelpFile, HelpId	S	Stored Procedure	Procedure Name[, param1..]									
Type	Description	Specifier																							
Q	QuickHelp	Name of the APPNOTE entity.																							
M	Message	Name of the MSG entity																							
H	WinHelp	HelpFile, HelpId																							
S	Stored Procedure	Procedure Name[, param1..]																							
<20,2>	Help specifier that is specific to the above Help Types.																								
<20,3>	<p>Button text. Defaults to "&Help".</p> <p>Note: Specify the AppNote, Message, HelpFile, or ProcName as it appears in the repository outliner. For example, the OINSIGHT.HLP file is registered as OINSIGHT, so specify the HelpFile as "OINSIGHT" (look in the outline under "General", "Windows Components", "Help Files")</p>																								
21	required	Boolean value specifying whether a response-type message requires a response; false allows nulls (default) while true doesn't.																							
22	beep	An integer value specifying a system beep. For more information, see MessageBeep in the Windows API.																							

Returns

Msg() return values are indicated in the following table.

Message Type	Return Value
BO	Returns null if OK clicked, Char(27) if escape pressed.
BOC	Returns null if OK clicked, Char(27) if escape pressed or cancel clicked.
BNY	Returns 1 if yes clicked, 0 if no clicked, Char(27) if escape pressed.
BNYC	Returns 1 if yes clicked, 0 if no clicked, Char(27) if cancel clicked or escape pressed.
BRC	Returns 2 if RETRY clicked, Char(27) if Cancel clicked or ESCAPE pressed
BAR	Returns 1 if ABORT clicked, 2 if RETRY clicked, Char(27) if ESCAPE pressed.
BARI	Returns 1 if ABORT clicked, 2 if RETRY clicked, 3 if IGNORE clicked, Char(27) if ESCAPE pressed.
B{list}	1, 2, or 3 for the first, second, or third button (from left to right), Char(27) if ESCAPE pressed.
R	Returns entered value if OK pressed, default response if CANCEL clicked or ESCAPE pressed.
RE	Returns entered value if OK pressed, Char(27) if CANCEL clicked or ESCAPE pressed.
RC, RCE	Returns the same as R and RE, except that the response is converted to uppercase.
U	Returns a D-type msgstructure that can be passed to a subsequent call of Msg() to take down the processing message.

Examples

Example 1

```
* request the user's password (and return it uppercased)

Def = ""

Def<MTEXT$> = "Enter your password:"

Def<MTYPE$> = "RCE"

Def<MICON$> = "?"

Def<MMASKINPUT$> = TRUE$

Password = Msg(@window, Def)

if Password = char(27) then

    * user escaped from the message

end else

    * user supplied password (which could be null) and pressed enter

end
```

Example 2

```
* display a message stored in the repository as "ERROR_MESSAGE"

Text = "An error occurred opening the table for processing."

Msg(@window, Text, "ERROR_MESSAGE")
```

Example 3

```
* display a message while batch processing

Def = ""

Def<MTEXT$> = "Processing..."

Def<MTYPE$> = "U"

* display the processing message and do the processing

MsgUp = Msg(@window, Def)

gosub BatchProcess

* take down the processing message

Msg(@window, MsgUp)
```

Example 4

```

/* display a gas gauge to show progress while processing 1000 orders */

$insert msg_equates

OrderCnt = 1000

Def = ""

Def<MCAPTION$> = "Processing Orders..."

Def<MTYPE$ > = "GC"

Def<MEXTENT$ > = OrderCnt

Def<MTEXTWIDTH$ > = 400

MsgUp = Msg(@window, Def)

for Order = 1 to OrderCnt

    /* process the order */

    * update the gauge and check if cancel was pressed

while Msg(@window, MsgUp, Order, MSGINSTUPDATE$)

next Order

/* we are done, take down the gas gauge */

Msg(@window, MsgUp)          ;* take down the gauge

```

Example 5

```

* logon processing with a batch screen

Def = ""

Def<MTYPE$> = "TA5"

* display 5-second splash-screen while performing login processing

Msg(@window, Def, "SPLASHSCREEN")

gosub Logon_Process

* allow the message to close if 5 seconds have elapsed

Yield()

gosub Setup

```