Productive Computing

Mail Manipulator

Functions Guide

Revised March 15, 2011

950 Boardwalk, Suite 205, San Marcos, CA 92078 • (760) 510-1200 • www.productivecomputing.com

TABLE OF CONTENTS

I. Introduction

Description:
The Mac Mail plug-in is a tool used to exchange data between FileMaker® and Mac Mail®. With this plug-in FileMaker users are able to bidirectionally exchange data between FileMaker and Mac Mail. More specifically you can import mail including attachments from Mac Mail into FileMaker, send mail with multiple attachments from FileMaker to Mac Mail, and move mail across folders within Mac Mail. These operations are accomplished using FileMaker function calls from within FileMaker calculations. These calculations are generally determined from within FileMaker "SetField," "SetVariable" or "If" script steps. For a list of the basic integration steps, please see the accompanying Developers Guide document.

Intended Audience:
FileMaker developers or persons, who have knowledge of FileMaker scripting, calculations and relationships as proper use of the plug-in requires that FileMaker integration scripts be created in your FileMaker solution.

Successful Integration Practices:
1) Read the Developer's Guide
2) Read the Functions Guide
3) Review our FileMaker demo and video tutorials
Demo and video tutorials can be found here: http://www.productivecomputing.com/plugins/mail-manipulator-id-53/
4) Familiarize yourself with Mac Mail

Error Handling:
Any of the plug-in functions may encounter an error during processing. When an error occurs during processing, immediately call the PCMM_GetLastError function in order to obtain a full description of the error or error number. This function returns the error message or error number associated with the last error in order to troubleshoot script or logic failures. Please see PCMM_GetLastError function description in the Functions's Guide and the "Handling Errors" section in the Developer's Guide for further clarification on how to properly trap for errors.

II. Function Descriptions

This section describes the functions that are available with the Mail Manipulator plug-in.

1) Error and Registration Related Functions

PCMM_GetLastError(type)

Purpose:
Returns a textual or numeric description of the last error the plug-in encountered. This gives the developer the option to trap errors as text or a number.

Dependencies:
Call at least one plug-in function

Parameters:

Parameter Name	Purpose	Values	Default Value
Type	Determines if the result is text or a number	Null, Text, Number	If null, default is text.

Return Values:
Returns the current error from the last function called.

Notes & Examples:
Example: PCMM_GetLastError("Text") could return "Plug-in Not Registered"

PCMM_GetOperatingMode

Purpose:
This returns the current mode of the plug-in.

Dependencies:
None

Parameters:
None

Return Values:
DEMO, LIVE, EXPIRED, UNREGISTERED or !!ERROR!!

Notes & Examples:
Plug-ins registered in DEMO mode will expire after thirty minutes in a session, or 30 days.

PCMM_Register(ServerName ; ServerPort ; ServerPage ; licenseID)

Purpose:
Registers the plug-in with Productive Computing's registration servers. Must call Register function once before using the plug-in.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
ServerName	This is the URL of Productive Computing's registration server.	"licensing.productivecomputing.com"	none
ServerPort	The port on which to access the server.	"80"	none
ServerPage	The page on the server where the registration code resides.	"/PCIReg/pcireg.asp"	none
LicenseID	The license provided with purchase of the plug-in.	"your license ID" or "DEMO-AM1"	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_Register("licensing.productivecomputing.com" ; "80" ; "/PCIReg/pcireg.asp" ; "SomeLicenseID")

PCMM_Version(type)

Purpose:
Used to identify the product and product version installed on a machine.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
type	Determines whether it returns just the version number, or the product name and number.	"short" or "long"	"short"

Return Values:
Product name and/or version, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_Version("long") would return "PC Mail Manipulator 1.0.0.1"

2) Mail Account, Folder and Record Specific Functions

PCMM_DeleteAllRecords

Purpose:
This function will delete all records from the current folder. This is a soft delete in that it will move the records to the account's trash folder.

Dependencies:
Must have opened a folder.

Parameters:
None

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
In the case of exchange accounts, it will move messages to Mac mail's trash folder, and they will no longer be visible on the exchange account.

PCMM_DeleteRecord(MacMailID)

Purpose:
Deletes a record in the currently opened folder. That record may either be the current record selected, or another record in the current folder, referenced by its MacMailID

Dependencies:
An account and folder must be currently open. If no parameter is supplied, a record must be currently open as well.

Parameters:

Parameter Name	Purpose	Values	Default Value
MacMailID	Selects which mail message to delete.	Mac Mail unique ID	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
This function cannot permanently delete the record because of Mac Mail limitations. The best that can be done is to move the mail to a trash folder on the host account. If no folder by the name "trash" exists on the host account, one must be created in order for deleted items to be properly relocated. Recognition of the trash folder name is case insensitive. However, in the case of exchange or Mobile me accounts, Mac Mail can delete all the records in the trash folder by a call to PCMM_EmptyAllDeletedItems

PCMM_Display(MacMailID)

Purpose:
Displays the current active message in Mac Mail opened with the PCMM_OpenRecord, PCMM_GetFirstRecord or PCMM_GetNextRecord.

Dependencies:
Must have called at least one open folder function followed by an open record function.

Parameters:

Parameter Name	Purpose	Values	Default Value
MacMailID	Selects which mail message to display.	Mac Mail unique ID	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_Display("SomeMacMailID")

PCMM_EmptyAllDeletedItems

Purpose:
This command will remove all items from the "Deleted Items" folder in Mac Mail. Keep in mind it will delete items you move there using the "DeleteRecord" command and it will also remove ALL objects that the user moved to this folder manually. This option should be used with caution.

Dependencies:
none

Parameters:
None

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
This function is limited with mail accounts like IMAP. It is only capable of deleting the email locally, and not on the server. So after the message is deleted, it will reappear when the account is synchronized again.

PCMM_GetAttachmentCount

Purpose:
Returns the number of attachments the currently opened item has. This function returns a text value. In order to perform numeric comparisons the result should be wrapped with the GetAsNumber function. [e.g. GetAsNumber (PCEM_GetAttachmentCount) > 100].

Dependencies:
Must have a record currently open.

Parameters:
None

Return Values:
The count, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_GetAttachmentName(Index)

Purpose:
Gets the name of the specified attachment to the currently opened Mac Mail Item.

Dependencies:
Must have a record currently open.

Parameters:

Parameter Name	Purpose	Values	Default Value
Index	The index of the attachment.	0 to one less than the number of attachments	None

Return Values:
The name of the attachment, otherwise an !!ERROR!!

Notes & Examples:
Attachments are indexed on a 1-n scheme, not 0-(n-1)
Example: PCMM_GetAttachmentName("1")

PCMM_GetFieldData(FieldName)

Purpose:
Extracts the value contained in the specified field, for the current record.

Dependencies:
Must have a record currently open.

Parameters:

Parameter Name	Purpose	Values	Default Value
FieldName	Determines which field to grab the data from in a given record.	To, Bcc, Cc, Body, Subject, Sender, Size, Sent, Received, Header, Flagged, Junked, Read, Forwarded, Redirected, Replied, ReplyTo	None

Return Values:
The string representation of the field value, otherwise an !!ERROR!!

Notes & Examples:
To: A comma separated list of the receivers
Bcc: A comma separated list of the blind receivers
Cc: A comma separated list of the copied receivers
Body: The body of the e-mail
Subject: The subject of the e-mail
Sender: The sender of the e-mail
Size: The size, in bytes, of the e-mail
Sent: The time stamp of when the e-mail was sent
Received: The time stamp of when the e-mail was received
Header: All the headers of the e-mail
Flagged: Boolean value, true if the e-mail is flagged
Junked: Boolean value, true if the e-mail is junk e-mail
Read: Boolean value, true if the e-mail has been read
Forwarded: Boolean value, true if the e-mail was forwarded
Redirected: Boolean value, true if the e-mail was redirected
Replied: Boolean value, true if the e-mail was replied to
ReplyTo: The e-mail address to send replies to

Example: PCMM_GetFieldData("Subject")

PCMM_GetFirstAccount

Purpose:
Opens the first Mac Mail account for reading and writing. Must call this function first after plug-in is registered.

Dependencies:
None

Parameters:
None

Return Values:
0 for success, otherwise "!!ERROR!!"

Notes & Examples:
none

PCMM_GetFirstFolder

Purpose:
Sets the first folder in a given account as the current operating folder.

Dependencies:
PCMM_GetFirstAccount, PCMM_GetNextAccount, PCMM_OpenAccount

Parameters:
None

Return Values:
A path to a folder, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_GetFirstRecord

Purpose:
Opens the first record in the currently opened folder. Sets it as the current working record.

Dependencies:
PCMM_GetFirstFolder, PCMM_GetNextFolder or PCMM_OpenFolder

Parameters:
None

Return Values:
The Mac Mail EntryID, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_GetNextAccount

Purpose:
Retrieves the next account.

Dependencies:
PCMM_GetFirstAccount or PCMM_OpenAccount

Parameters:
None

Return Values:
Name of the account or "End" if last account. Or !!ERROR!!

Notes & Examples:
none

PCMM_GetNextFolder

Purpose:
Sets the next folder as the working folder.

Dependencies:
PCMM_GetFirstFolder or PCMM_OpenFolder

Parameters:
None

Return Values:
A path to a folder, “End” if there are no more folders, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_GetNextRecord

Purpose:
Opens the next record in the currently opened folder.

Dependencies:
PCMM_GetFirstRecord

Parameters:
None

Return Values:
‘End’ if no more records or the MacMailID of the next record, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_GetRecordCount

Purpose:
Returns the number of available records in the currently opened folder. This function returns a text value. In order to perform numeric comparisons the result should be wrapped with the GetAsNumber function. [e.g. GetAsNumber (PCEM_GetRecordCount) > 100].

Dependencies:
PCMM_GetFirstFolder, PCMM_GetNextFolder or PCMM_OpenFolder

Parameters:
None

Return Values:
The number of records, otherwise an !!ERROR!!

Notes & Examples:
none

PCMM_MoveRecord(FolderPath)

Purpose:
Moves the currently opened Mac Mail item from its current folder to another folder.

Dependencies:
PCMM_GetFirstRecord, PCMM_GetNextRecord or PCMM_OpenRecord

Parameters:

Parameter Name	Purpose	Values	Default Value
FolderPath	The path of the destination folder. The path must be the full path retrieved with PCMM_GetFirstFolder or PCMM_GetNextFolder.	"some folder|sub folder"	None

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_MoveRecord("Inbox")

PCMM_NewFolder(Name)

Purpose:
Creates a new folder with the specified path. Just a name with no path will create a folder in the root of the account folder directory.

Dependencies:
PCMM_GetFirstAccount, PCMM_GetNextAccount or PCMM_OpenAccount

Parameters:

Parameter Name	Purpose	Values	Default Value
Name	The name of the folder to create.	some folder name	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_NewFolder("SomeFolderName")

PCMM_OpenAccount(Name)

Purpose:
This opens a specified account based upon the name provided.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
Name	The name of the account to be opened.	some account name	None

Return Values:
0 for success, or an !!ERROR!!

Notes & Examples:
Example: PCMM_OpenAccount("gmail")

PCMM_OpenFolder(FolderPath)

Purpose:
Opens the specified Folder for editing.

Dependencies:
PCMM_GetFirstAccount, PCMM_GetNextAccount or PCMM_OpenAccount

Parameters:

Parameter Name	Purpose	Values	Default Value
FolderPath	The path to the folder from the currently opened account.	some folder path	none

Return Values:
The Mac Mail EntryID for the Folder, otherwise an !!ERROR!!

Notes & Examples:
Folder names are case sensitive.
Example: PCMM_OpenFolder("My Folder Name/My Sub Folder Name")

PCMM_OpenRecord(MacMailID)

Purpose:
Opens the specified record for editing.

Dependencies:
PCMM_GetFirstFolder, PCMM_GetNextFolder or PCMM_OpenFolder

Parameters:

Parameter Name	Purpose	Values	Default Value
MacMailID	The Mac Mail ID of the item to open.	Mac Mail unique ID	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_OpenRecord("SomeMacMailID ")
The mail id is retrieved with the functions PCMM_GetFirstRecord or PCMM_GetNextRecord.

PCMM_SaveAttachment(FilePath ; Index)

Purpose:
Saves the attachment specified by its index, to the provided path.

Dependencies:
Must have a record currently open.

Parameters:

Parameter Name	Purpose	Values	Default Value
FilePath	The full path that you would like to save the attachment to.	some file path	none
Index	The index of the items attachment.	some index	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
The indices of the attachments always start at 1. So if there is one attachment, then its index is 1. If there are ten attachments, then their indices are 1 through 10. The PCMM_SaveAttachment function will overwrite an existing file with the same name. Specify attachment paths and names carefully.
Example: PCMM_SaveAttachment("/Drive/Users/johndoe/Desktop"; 1)

PCMM_SendMail(To ; Cc ; Bcc ; Subject ; Body ; Attachments)

Purpose:
Use to send mail in a single step. This function sends an e-mail message to the intended recipients with the given subject, body, attachments and formatting options. All parameters except Subject and Body can contain multiple values.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
To	The recipients of the e-mail separated by a comma ','	some email	none
Cc	The carbon copy recipients separated by a comma ','	some email	none
Bcc	The blind carbon recipients separated by a comma ','	some email	none
Subject	The subject of the message	some subject	none
Body	The Body of the message	some body	none
Attachments	Any attachments for the message, specified by the local mac path with '|' separated values.	some mac path	none

Return Values:
0 for success, otherwise an !!ERROR!!

Notes & Examples:
Example: PCMM_SendMail("joe@somecompany.com" ; "frank@anothercompany.com" ; "manager@somecompany.com" ; "New Release" ; "A new product was just released ... please contact us!" ; "“/SomeHD/Users/Paul/Documents/SomeFile.pdf”")

PCMM_SetFolderDelimiter(Delimiter)

Purpose:
This function sets the character string that will delimit folders and subfolders.

Dependencies:
The plug-in must be registered, and you must provide a string of characters.

Parameters:

Parameter Name	Purpose	Values	Default Value
Delimiter	The string that separates folder names in the Mail Manipulator Plug-in.	some string	None

Return Values:
0 on success, otherwise an !!ERROR!!

Notes & Examples:
The preset value is a "|"
Example: PCMM_SetFolderDelimiter("l")
The the folders would be separated like: "ParentFolderlChildFolder"
Notice the lack of spaces. If you include spaces in the function call, such as PCMM_SetFolderDelimiter(" l "), then those spaces will be apart of the delimiter and the folders would be separated like:"ParentFolder l ChildFolder"

III. Available Mac Mail Fields

Mac Mail has various available areas fields when sending and importing mail. In order to send mail, you will call the PCMM_SendMail function and specify the specific fields to "set." In order to import or "get" a Mac Mail field you must call the PCMM_GetFieldData function and specify the field you would like to "get."

All available fields are listed below. If there is a missing field that you would like to access, then please email a request to support@productivecomputing.com

Available Mail Fields

Mail Fields represent a field in a mail message in Mac Mail.

Method	**Field Name**	**Acceptable Values/Notes**
Get/Set	To	A comma separated list of the receivers
Get/Set	Bcc	A comma separated list of the blind receivers
Get/Set	Cc	A comma separated list of the copied receivers
Get/Set	Body	The plan text body of the e-mail
Get/Set	Subject	The subject of the e-mail
Get	Sender	The sender of the e-mail
Get	Size	The size, in bytes, of the e-mail
Get	Sent	The time stamp of when the e-mail was sent
Get	Received	The time stamp of when the e-mail was received
Get	Header	All the headers of the e-mail
Get	Flagged	Boolean value, true if the e-mail is flagged
Get	Junked	Boolean value, true if the e-mail is junk e-mail
Get	Read	Boolean value, true if the e-mail has been read
Get	Forwarded	Boolean value, true if the e-mail was forwarded
Get	Redirected	Boolean value, true if the e-mail was redirected
Get	Replied	Boolean value, true if the e-mail was replied to
Get/Set	ReplyTo	The e-mail address to send replies to

IV. Contact Us

Successful integration of a FileMaker plug-in requires the creation of integration scripts within your FileMaker solution. A working knowledge of FileMaker Pro, especially in the areas of scripting and calculations is necessary. If you need additional support for scripting, customization or setup (excluding registration) after reviewing the videos, documentation, FileMaker demo and sample scripts, then please contact us via the avenues listed below.

Phone: 760-510-1200
Email: support@productivecomputing.com
Forum: www.productivecomputing.com/forum

Please note assisting you with implementing this plug-in (excluding registration) is billable at our standard hourly rate. We bill on a time and materials basis billing only for the time in minutes it takes to assist you. We will be happy to create your integration scripts for you and can provide you with a free estimate if you fill out a Request For Quote (RFQ) at www.productivecomputing.com/rfq . We are ready to assist and look forward to hearing from you!