Comm-Unity Networking Systems

CNS

SMTPit
A Plug-in for FileMaker Pro

- Use normal email fields such as To, CC, BCC, From, Subject, and Body.
- Create HTML Emails complete with inline graphic images.
- Attach files to your emails.
- Assign text or html files on your hard drive for the body of a message.

Contents

Index of Figures

Introduction

Sending Email from within FileMaker Pro puts you in the driver's seat for complete communication. SMTPit allows you to send Email with several custom made features that just are not available in other products. With SMTPit, mail merges are easy with our header and footer implementations. Imagine having a custom introduction to each of your customers aimed directly at them, calling them by name. From mailing lists to newsletters, SMTPit makes it simple, giving you the ability to have custom headers and footers for each Email. SMTPit also works well with WebCompanion solutions. Imagine sending personalized thank you notes to every web visitor that fills out one of your web forms. You could even tailor those thank you notes with product details based on the web visitor's interests. SMTPit also has the ability to send HTML only emails, or multipart/alternative Text & HTML emails for those email clients that do not understand HTML content. Plus, you can include inline HTML images to provide graphic logos or other images in your HTML emails. Since SMTPit is so flexible, there are countless solutions. Imagination is the only limitation.

Features

- Use normal email fields such as To, CC, BCC, From, Subject, and Body.

- Unique Header and Footer functions.

- Create HTML Emails complete with inline graphic images.

- Attach files to your emails.

- Assign text or html files on your hard drive for the body of a message.

- Include extra Email Headers, such as "X-Mailer: SMTPit".

Installation & Configuration

Installation

Installing SMTPit is very easy. If FileMaker Pro is open, then close it. Next, locate the folder on your hard drive that contains your FileMaker Pro application. For Macintosh, this is usually in the Applications folder; while on Windows, it is usually in the Program Files folder. Next to the FileMaker Pro application, you should see a folder named FileMaker Extensions (on Macintosh; See Figure 1) or System (on Windows; See Figure 2). Copy the plug-in file into this folder to install it into FileMaker Pro.

Figure 1. Installing on Macintosh

Figure 2. Installing on Windows

After installing the plug-in as described above, you can open FileMaker Pro again and set the default perferences if you wish. To do this, go to Edit, Preferences, Application, click on the Plug-ins tab, and double-click the SMTPit plug-in.

Basics

Once the Configuration Dialog is open, click the Basics tab (See Figure 3) where you can set the following values. **SMTP Host** is where you enter the domain name or IP address of your mail server. **From** is where you put your email address. **Subject** is where you can put a default Subject for emails where you do not specify a Subject. **Priority** is where you set a default Priority.

SMTPit will refer to the default settings when you do not specifically set values in your scripts. In other words, if you do not define a from email address in your SMTPit related script, SMTPit will use the Default From address.

SMTPit waits for certain amount of time for your SMTP host to respond. By default, this is set to 30 seconds. However, you can adjust the setting to fit your needs by entering a different value in the Connection Timeout.

To be compliant with the SMTP protocol internet standard, SMTPit wraps all long lines down to a specified length before transferring your email. SMTPit adds the appropriate Email Headers for modern email clients to "unwrap" those lines back to their original length, so you will probably never know the difference. However, if you do need to change how many characters SMTPit leaves on a line, you can adjust the SMTP Line Break Column setting. The default setting is 80.
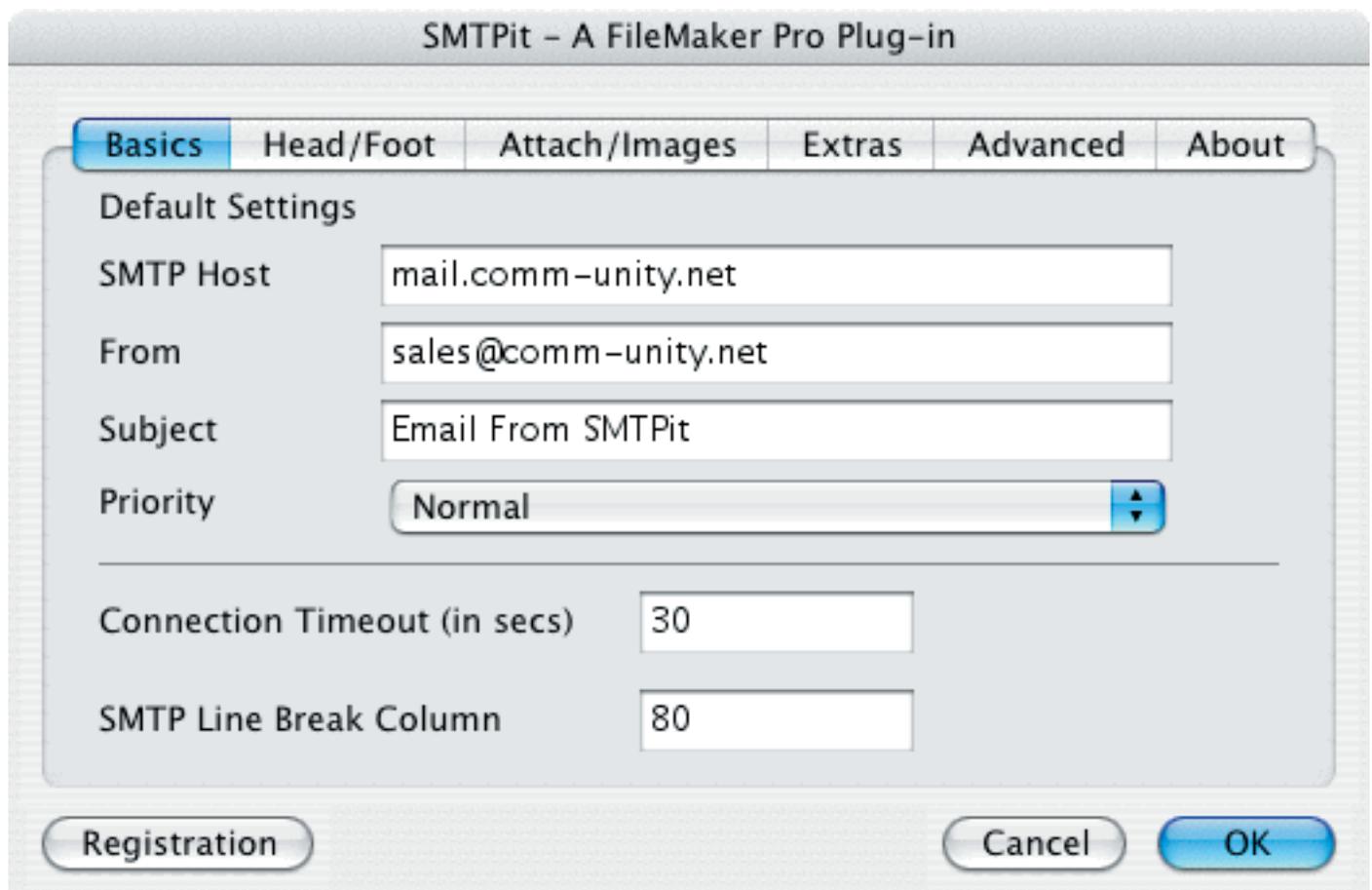
Figure 3. Basics tab

Headers and Footers

SMTPit has the unique ability to add headers and footers to the Plain Text Body and HTML Body of email messages. To enter a default header or footer simply click on one of the "Header" or "Footer" tabs (See Figure 4) in the Configuration Dialog. Then simply click in the edit field to enter a default header or footer. Like most SMTPit functions, you can replace the default values in your current scripts with values from fields in your database. You can also use the related append functions to add to the default headers and footers.

Because of this function, you can easily insert personal information into your email messages. You could have a default header and footer, and use the SMTP-HeaderAppend function to add a personal message or name to each email message. Another solution is to have a header or footer field in your database that is a calculation. With this scenario you could dynamically insert product information from your database into each email. Keep in mind that SMTPit also has the SMTP-BodyAppend and SMTP-HTMLBodyAppend functions that can be integrated into such a solution.
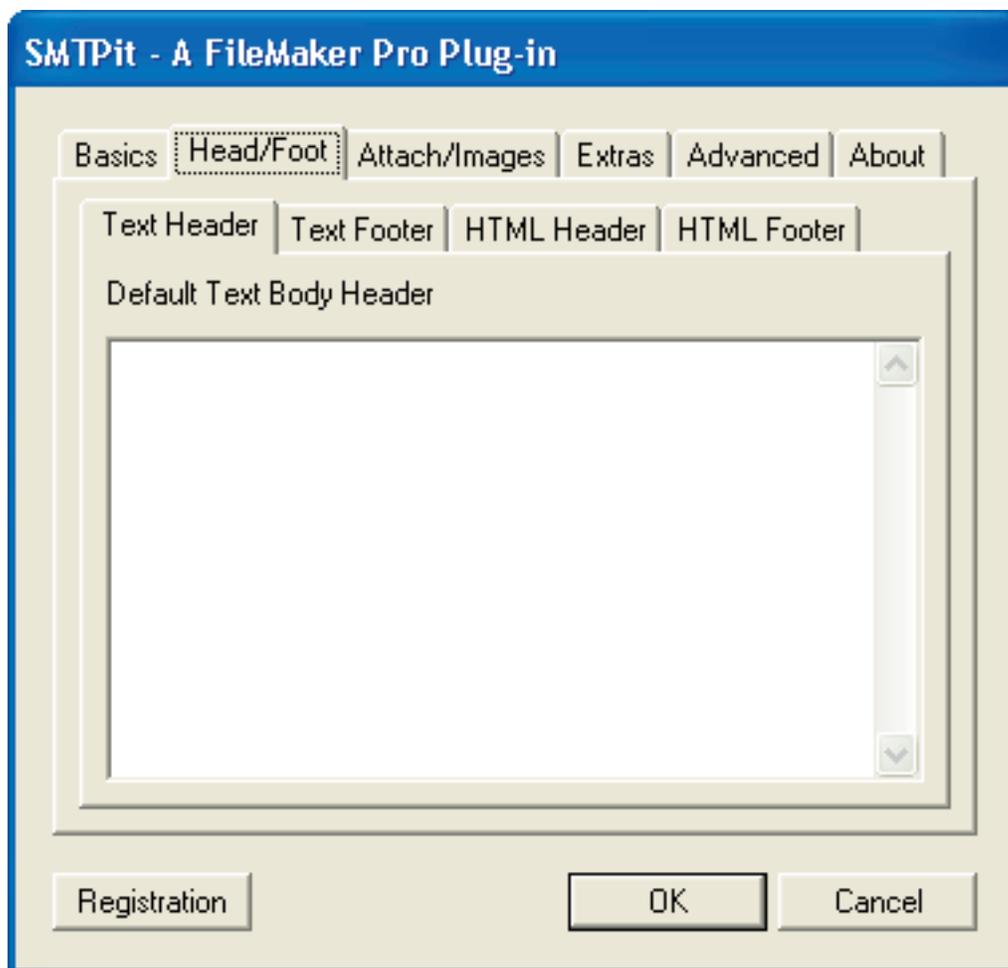
Figure 4. Header and Footer tab

Attachments

SMTPit has the ability to send attachments either using the default attachments set in the Configuration Dialog (See Figure 5) or via six functions: SMTP-AttachAssign, SMTP-AttachAppend, SMTP-DlgAttachAssign, SMTP-DlgAttachAppend, SMTP-FileNameAcquire, and SMTP-PathToDBAcquire. To set a default attachment, one that is always sent, click on the Add button on the Attachments tab. After you click the Add button, a standard open file dialog will pop up which will allow you to navigate to your desired attachment. Once you have found the file you want to attach, simply click the open button. To add additional default attachments, simply repeat the process above.

As mentioned above SMTPit has six functions related to attachments. The SMTP-AttachAssign and the SMTP-AttachAppend functions require a complete path to a valid file. To make sure you have the correct path, use the SMTP-FileNameAcquire function, which will present you with a standard open file dialog to locate the desired file. (Note: SMTP-FileNameAcquire only returns a valid path. It does not use the selected file as an attachment.) You can also use the SMTP-PathToDBAcquire function to return a complete path to the folder on your hard drive that contains your database. This is useful if you have attachments in the same folder as your database. The SMTP-DlgAttachAssign and the SMTP-DlgAttachAppend present you with a standard open file dialog to choose an attachment just like the Configuration Dialog mentioned in the last paragraph. For more information on these functions, refer to the SMTPit External Functions Reference section of this documentation or refer to the SMTPit_ Functions database that came with the SMTPit archive.
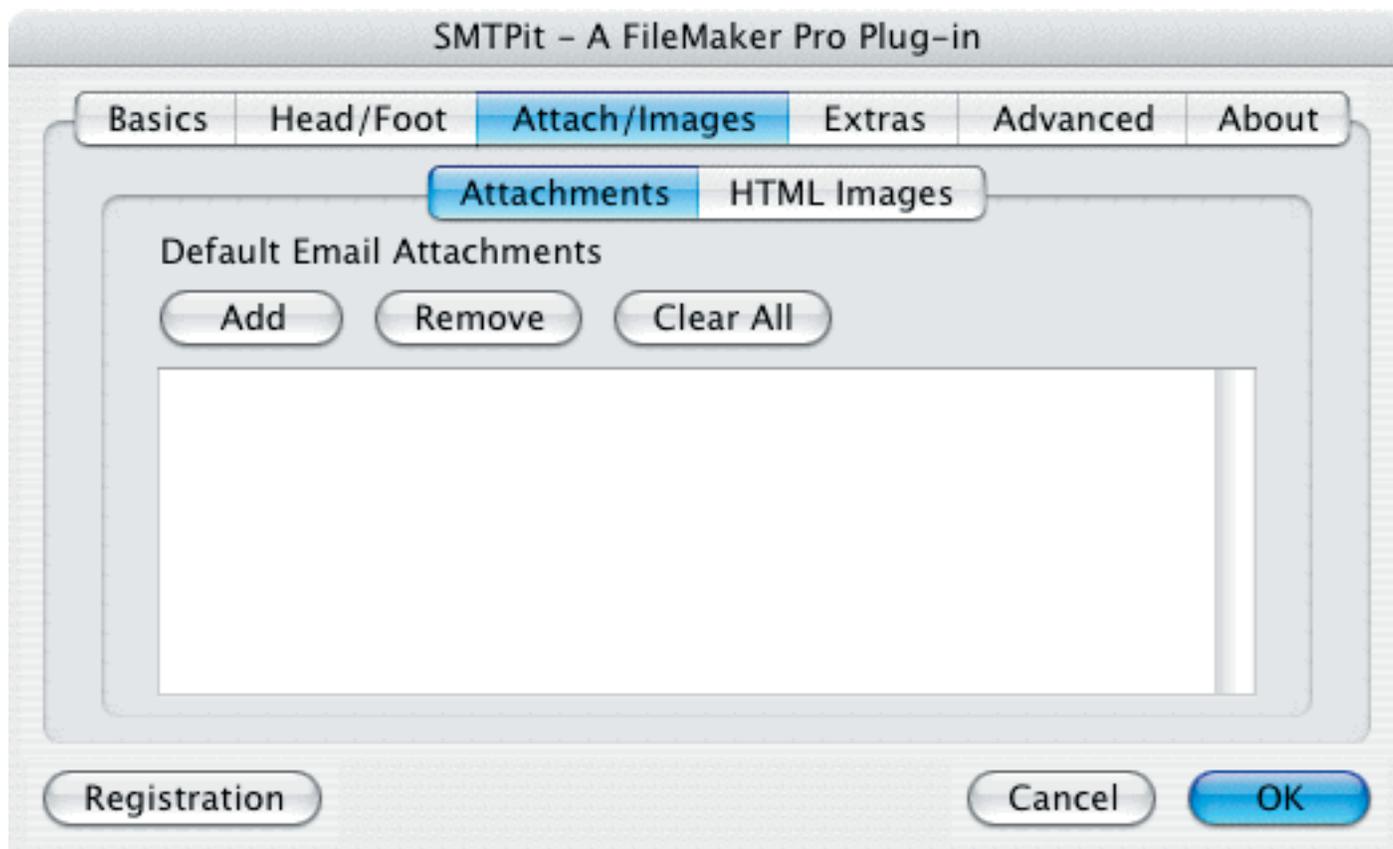
Figure 5. Attachments tab

HTML Images

With the ability to send HTML emails, comes the ability to send Inline HTML Images for display in the HTML emails. You can either assign default HTML Images in the Configuration Dialog (See Figure 6), or add HTML Images via four functions: SMTP-HTMLImageAssign, SMTP-HTMLImageAppend, SMTP-DlgHTMLImageAssign, and SMTP-DlgHTMLImageAppend. To set a default HTML Image, one that is always sent, click on the Add button on the HTML Images tab. After you click the Add button, a standard open file dialog will pop up which will allow you to navigate to your desired HTML Image. Once you have found the image you want, simply click the open button. To add additional default HTML Images, simply repeat the process above.

As mentioned above, SMTPit has four functions related to HTML Images. The SMTP-HTMLImageAssign and the SMTP-HTMLImageAppend functions require a complete path to a valid file. Like the related Attachment functions, you can make sure you have the correct path by using the SMTP-FileNameAcquire function, which will present you with a standard open file dialog to locate the desired HTML Image. (Note: SMTP-FileNameAcquire only returns a valid path. It does not use the selected file as an HTML Image.) You can also use the SMTP-PathToDBAcquire to return the complete path to the folder on your hard drive that contains your database, or SMTP-PathToFMAcquire to return the complete path to the folder on your hard drive that contains FileMaker . This is useful if you have HTML Images in the same folder as your database or FileMaker.
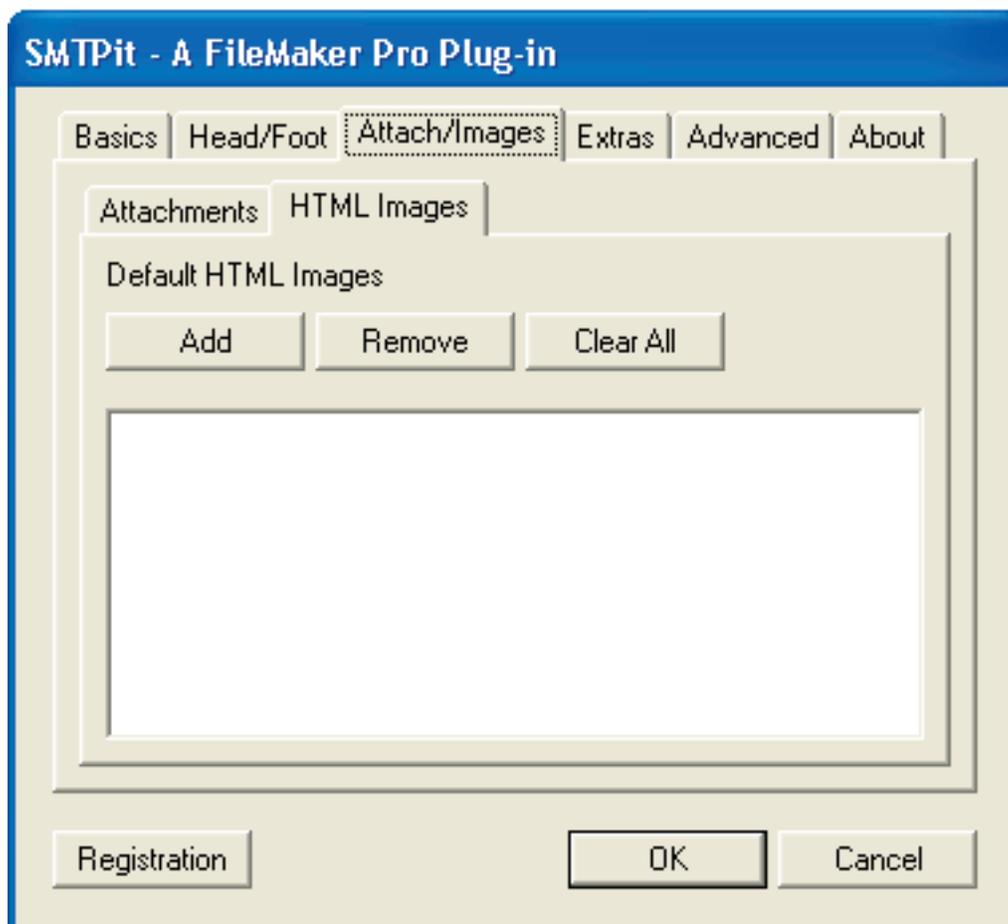
Figure 6. HTML Images tab

The SMTP-DlgHTMLImageAssign and the SMTP-DlgHTMLImageAppend functions present you with a standard open file dialog to choose an HTML Image just like the Configuration Dialog mentioned in the last paragraph. For more information on these functions, and on using inline HTML Images, refer to the HTML Image Example Explained and the SMTPit External Functions Reference sections later in this documentation, or refer to the SMTPit Functions database that came with the SMTPit archive.

Email Headers

SMTPit also has the ability to insert extra Email Headers into your email messages. For example, you can identify your company by creating an Email Header with the name, "X-Company", and the value being the name of your company. When the recipient gets your email message they will see "X-Company: Your Company Name" at the top of the email. It is important that you do not have any spaces in the Name of the header due to Email Header specifications, however, you can have spaces in the value. You can also use Email Headers to track different email promotions or use in conjunction with POP3it to run a script based on an Email Header.

To add a Default Email Header simple click on the Add button as shown below (See Figure 7), then type in your desired header name in the name field with the desired value in the value field. You can also add email headers dynamically by using the SMTP-EmailHeaderAppend function. The default Email Headers you define in the Configuration Dialog will be appended to the header of the email before any headers added by the SMTP-EmailHeaderAppend function.

Figure 7. Email Headers tab

Authentication

SMTPit supports the following Authentication Types (in increasing level of security): None (off), Plain, Login, CRAM-MD5. To set SMTPit to use authentication support, set the Type to something other than None, and fill in the Username and Password fields with the settings for your email account (See Figure 8).
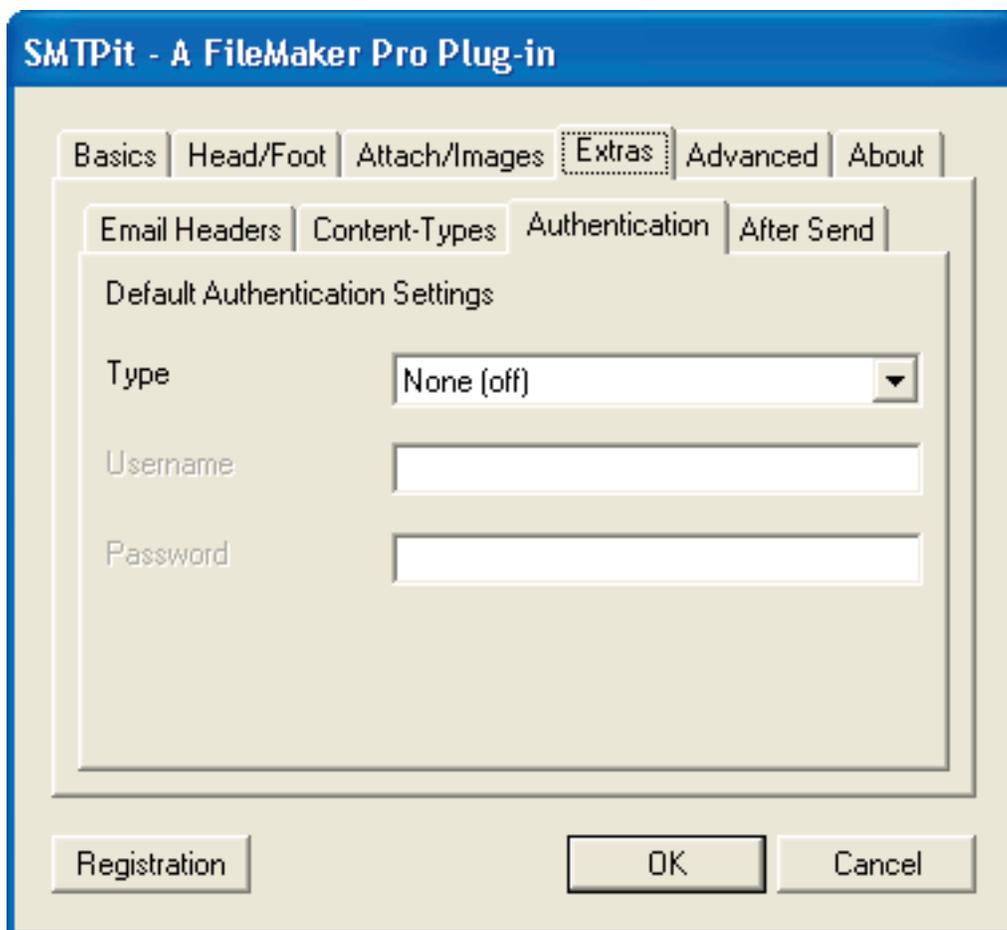
Figure 8. Authentication tab

After Send

The After Send settings (See Figure 9, Next Page) will determine what action SMTPit takes after sending each email. If you would like SMTPit to pop up a dialog window letting you know that your email message was sent, check the Popup "Email Sent Successfully" Dialog checkbox. This option is often helpful when debugging your scripts. You can also tell SMTPit to pop up this dialog on a per email basis with the optional parameters to the SMTP-Send function. For more information on how to do this, see the SMTPit External Function Reference later in this document, or the SMTPit Functions database that came in the SMTPit archive.

You also have the option to clear the following fields: **To, From, Subject, CC, BCC, Text Body, Text Header, Text Footer, Attachments, Email Headers, HTML Body, HTML Header, HTML Footer, HTML Images, Priority, Port, AuthType, Username, and Password**. Fields that are checked will clear after each email message is sent. Fields that are not checked retain their set values after each email message is sent.

To better explain this, here is an example scenario: You have a database of contacts and you want to send a form letter to each of those contacts. The body of each of these emails is exactly the same, but you want the subject to say "Dear Frank", where "Frank" is the name of the contact. To do this, put "Dear " in the Subject field on the Basics tab, and on the After Send tab, check off To and Subject, and leave everything else unchecked. Next, make a script for your database that first assigns the body of the message and then goes into a loop to send all the messages. Inside this loop, you only have to assign the "To" field, and use SMTP-SubjectAppend to add on the contact's name. Everything else, including the body, will be retained after each email is sent for use in the next email.
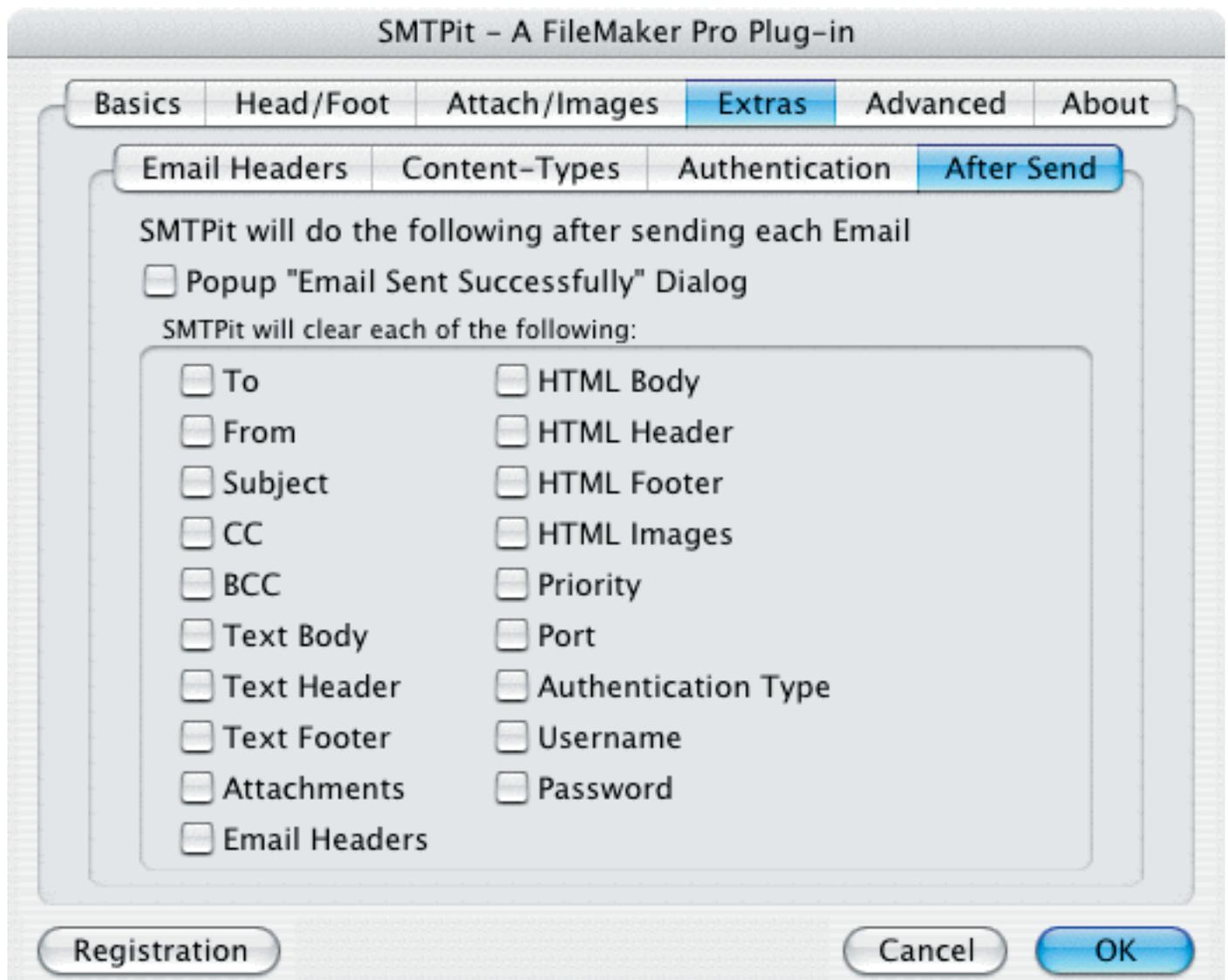
Figure 9. After Send tab

About

The About Tab simply reports which version of SMTPit that you are using. You can also use the SMTP-Version function to bring up the Configuration Dialog by passing the function the string "CONFIGURE" or the string "ABOUT". To make sure that you have the most resent version of SMTPit, please visit our website (http://www.smtpit.com/).
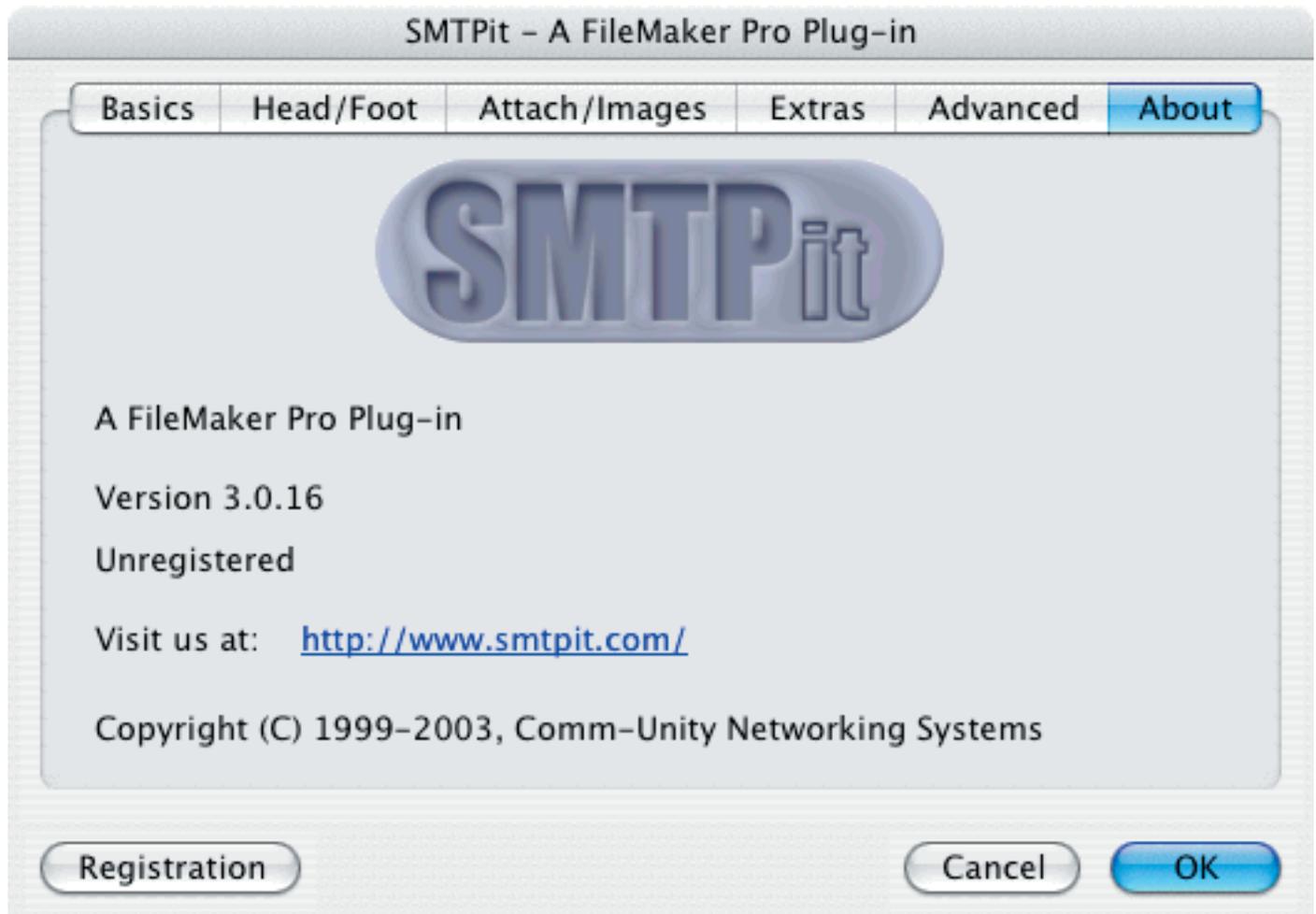
Figure 11. About tab

Registration

If you find SMTPit useful and would like to register it, you can purchase a license to use it from our secure online purchase form. Simply go to http://www.cnsplug-ins.com/ and click on the Purchase link. Once you have purchased a license to use SMTPit, you will be sent a registration code to register your copy of the plug-in. To do that, simply open any of the example databases and press the "Register SMTPit" button. and enter in your registration information. You can also register by opening the configuration dialog, and clicking the Registration button.

Quick Start Example Explained

The Quick Start Example database is a very simple database that should help you understand the very basics of sending email with SMTPit. This database contains the following six fields:

Field Name	Field Type
SMTP Host	Text
To Email Address	Text
From Email Address	Text
Subject	Text
Body	Text
SMTPit Result	Text

The first five fields should be self-explanatory; you should be able to fill them out with your information. The sixth field, SMTPit Result, will hold the results of the SMTPit External Functions. You do not need to put anything in this field because when you press the Send button, the SMTPit Result field will be filled in by SMTPit.

Let us look at the scripts in this database. Go to the Script menu and select ScriptMaker... Select the Send (with fields) script, and press the Edit button. The Script Definition for "Send (with fields)" dialog should open. In the big text area on the right, select the Set Field Script Step. Near the bottom of the dialog, some buttons should have appeared when you selected the Set Field Script Step. Press the Specify... button. Now the Specify Calculation dialog is open. You should see several External() functions in the large text area:

        External("SMTP-ClearAll", "")

This tells SMTPit to clear any previous settings it may have. It is a good idea to always start with this function so that you can ensure you are starting with a clean slate.

        External("SMTP-HostAssign", SMTP Host)

This tells SMTPit the mail server host to connect to using the value you specified in the SMTP Host field.

        External("SMTP-ToAssign", To Email Address)

This tells SMTPit the to email address of the person you are sending email to, using the value you specified in the To Email Address field.

        External("SMTP-FromAssign", From Email Address)

This tells SMTPit the From Email Address.

```
External("SMTP-SubjectAssign", Subject)
```

This tells SMTPit the subject of the email you are creating.

```
External("SMTP-BodyAssign", Body)
```

This tells SMTPit the content of the email.

```
External("SMTP-Send", "")
```

And finally, this tells SMTPit to send the email you just created.

That is all there is to it. If you fill in those five fields, SMTP Host, To Email Address, From Email Address, Subject, and Body, and then press the Send button, then SMTPit will send an email based on the values you put in those fields. If SMTPit has an error with something you entered into the fields, or if it has a problem connecting to your mail server or sending your email, you can look at the SMTPit Result field. This field will contain the results of each of the functions in the Set Field Script Step in the Send (with fields) script.

You can also look at the Send (with hard-coded values) script to see how you can enter in literal values into the SMTPit External Functions. You can mix and match literal values with fields as well. Now that you know how simple it is to send email with SMTPit, you can add a few fields and a script or two to your own databases, and have email sending capabilities in no time!

Simple Mailer Example Explained

This section of the documentation will explain the Send - Text script of the Simple Mailer Example database. This script is a little more advanced than the script in the Quick Start Example database. It should give you a few more ideas on how to use some of the more advanced features of SMTPit, while not getting overly advanced. There are fourteen fields in this database that are used by the Send - Text script:

Field Name	Field Type
SMTP Host	Global (Text)
Authentication Type	Global (Text)
Username	Global (Text)
Password	Global (Text)
From Email Address	Global (Text)
To Email Addresses	Text
CC Email Addresses	Text
Subject	Text
Priority	Text (Auto Enter: "Normal")
Attachments	Text
Text Body	Text
Text Signature	Global (Text)
Setup Result	Text
Send Result	Text

Note that not all fields are text like in the Quick Start Example database. The SMTP Host, Authentication Type, Usename, Password, From Email Address, and Text Signature fields are global fields which will allow you to have the same value for each record in your database. An alternative way to do this would be to use the default settings in the Configuration Dialog as discussed earlier in this documentation. The Priority text field is a little different in that it has an Auto Entered value of "Normal". This will set the priority to it's default value, "Normal", when you create a new record.

Send - Text Script

Now that you know the fields in the database, let us look at the "Send - Text" script. (It has been slightly modified for this documentation to exclude a few steps that do not pertain to sending email with SMTPit.)

Comment ["Check to make sure there are recipients."]

This step contains a simple Comment Script Step. Comment Script Steps are useful for making notes about what a particular part of the script is doing so that if you ever need to modify your script, you can easily find the part you need to modify. You can also use Comment Script Steps to remind yourself why you did something a certain way or what parts you should not touch so that you do not break something.

If ["IsEmpty(To Email Addresses) and IsEmpty(CC Email Addresses)"]

This step contains an If Script Step. An If Script Step can be used to check a value and then take a course of action based on the result. In this instance, we are checking to see if the To Email Addresses and CC Email Addresses fields are empty or not.

Show Message ["This email is missing recipients. Please enter an email address in the
    To field before pressing the Send button."]
Exit Script

If they are empty, then in this step we inform the user with a Show Message Script Step that they need to put an email address in the To Email Addresses Field, and then we exit the script. An Exit Script Step tells FileMaker Pro to stop executing the script and return control to the user. FileMaker Pro will not execute any more Script Steps after an Exit Script Script Step.

Comment ["Set up the email."]

If there is at least one email address in the To Email Addresses or CC Email Addresses fields, then the Show Message and Exit Script Steps will be skipped and this will be the next Line, which is another Comment Script Step.

Set Field ["Setup Result",
    External("SMTP-ClearAll", "") & "¶" &
    External("SMTP-HostAssign", SMTP Host) & "¶" &
    External("SMTP-AuthTypeAssign", Authentication Type) & "¶" &
    External("SMTP-UsernameAssign", Username) & "¶" &
    External("SMTP-PasswordAssign", Password) & "¶" &
    External("SMTP-FromAssign", From Email Address) & "¶" &
    External("SMTP-ToAssign", To Email Addresses) & "¶" &
    External("SMTP-CCAssign", CC Email Addresses) & "¶" &
    External("SMTP-SubjectAssign", Subject) & "¶" &
    External("SMTP-PriorityAssign", Priority) & "¶" &
    External("SMTP-AttachAssign", Attachments) & "¶" &
    External("SMTP-BodyAssign", Text Body) & "¶" &
    External("SMTP-FooterAssign", Text Signature)]

The next Line is a Set Field Script Step. Remember in our Quick Start Example database, all of our interaction with SMTPit happened in the calculation field of a Set Field Script Step. Well, this is where the "Setup" SMTPit functions are for this script.

The first two lines should be familiar to you from the Quick Start Example database. The next three lines set the Authentication Type, Username, and Password for your email account on the mail server. If your mail server requires you to log in before sending email, then you can set values in these fields in the Preferences layout. If you set the Authentication Type to None and leave the Username and Password fields blank, then these External Functions will not do much, but they will not return errors for giving them blank values.

The SMTP-FromAssign and SMTP-ToAssign functions should also look familiar from the Quick Start Example. SMTP-CCAssign works exactly like SMTP-ToAssign, except that it sets the CC or Carbon Copy Email Addresses instead of the To Email Addresses. Any email addresses you have in the CC Email Addresses field will be sent a copy of the email that you are sending.

SMTP-SubjectAssign looks the same as the Quick Start Example. The SMTP-PriorityAssign function sets the priority of the email. Most modern email clients can recognize the priority of email and will highlight your email as being more or less important than other emails around it. SMTP-AttachAssign sets the File Attachments you want to send with your email. You can use the Add button next to the Attachments field on the layout to add extra attachments, or you can type them in yourself. Just remember that you need the Full Path and FileName of the attachment on your hard drive so that SMTPit can find it. If you do not need to send any files with your email, you can leave the Attachments field blank, and SMTPit will just ignore your SMTP-AttachAssign function. SMTP-BodyAssign looks like the Quick Start Example, but the last line shows another new function. This time we are using the SMTP-FooterAssign function to attach an Email Signature. SMTPit will put anything you place in the "Footer" onto the end of your email. So, after you fill in all of your email content in the Body field, SMTPit will place your Email Signature onto the end for you before it sends the email.

```
If ["PatternCount(Setup Result, "ERROR:")"]
```

Next we go back to our Send - Text script and look at the next line. This line uses another If Script Step and the PatternCount Text Function to search the Setup Result field for the word "ERROR:". The Setup Result field was filled in by all of your External Functions in the previous Set Field Script Step. So, after FileMaker Pro calls all of the External Functions for us, we can then look into the field and see if there were any errors. If there are errors, then we go to the Show Message Script Step. If there are no errors, then we go to the Comment Script Step.

```
Show Message ["There was an error setting up your email. Please check the
        Troubleshoot layout to determine the cause."]
```

This Script Step uses another Show Message Script Step to inform the user that there was an error and that they need to look at the Setup Result field to determine why there was an error. FileMaker Pro would then jump to the line after the last End If Script Step, if there was a line after the last End if Script Step. Since there is not, FileMaker Pro stops executing the script and returns control to the user. The user could then inspect the Setup Result field for the error.

```
Comment ["Send the message."]
```

If there were no errors, then we go to this line for our next Script Step. This line contains another Comment Script Step telling us that we are going to now send the message.

```
Set Field ["Send Result", "External("SMTP-Send", "")"]
```

This line uses another Set Field Script Step to call the SMTP-Send External Function. In this documentation, we have simplified the calculation for this Set Field Script Step to just contain "External("SMTP-Send", "")". This should look familiar from the Quick Start Example database, and like in the Quick Start Example database, this tells SMTPit to send the email. The actual script in the Send - Text script contains the optional SMTP-Send parameter,

Transcript, and if you are interested in what that does, please see the SMTP-Send documentation in the SMTPit External Functions Reference later in this document or in the SMTPit Functions database.

> If ["PatternCount(Send Result, "Success")"]

In this If Script Step, we are checking to see if the SMTP-Send function returned successfully. If SMTPit successfully sent the email, then the Send Result field will contain the value "Email Sent Successfully." Since it is possible that future versions of SMTPit might return a different success result, or might contain more information, it is a good idea to use the PatternCount Text Function to search for the word "Success". This is much safer than using an If Script Step Like this:

> If ["Send Result = "Email Sent Successfully."]
> ...
> End If

If you compare a value to a literal string like "Email Sent Successfully.", then you are setting yourself up for disaster if that literal string ever changes. This is why we encourage you to use the PatternCount Text Function to test the result of any plug-in External Function, not just ours.

> Show Message["Your email was sent!"]

If the If Script Step found the word "Success" in the Send Result Field, then this line is executed. This line uses another Show Message Script Step to inform the user that the email was sent successfully. Note that the actual script in the database does not include this Script Step.

> Show Message["There was an error sending your email. Please check the Send Result field and/or the Troubleshoot layout to determine the cause.

If the If Script Step did not find the word "Success", then this line is executed, which uses a Show Message Script Step to inform the user that their email was not successfully sent and that they should look at the Send Result field to determine why.

Note that the documentation here differs from the actual Send - Text script in the database. The documentation shows a much simpler set of Script Steps instead of the layout switching and other Script Steps that go on in the actual database that don't really have anything to do with SMTPit.

Add Attachment Script

This script simply calls the SMTP-FileNameAcquire function, which brings up a standard open file dialog that allows you to locate a file on your hard drive that you want to attach. This function does not actually attach the file to the email, but instead returns a full path and filename to the file and places it in the Attachments field. When one of the Send - XXX scripts is called, the contents of the Attachments field is sent to the plug-in so that it can attach the files.

Well, now you have seen a slightly more advanced script for sending email with SMTPit. We hope that you are starting to see that SMTPit is easy to use and flexible enough to do just about anything you want with your email

Mass Mailer Example Explained

The Mass Mailer Example database shows how to exploit several of the unique features of SMTPit. The most important feature that makes this Mass Mailer Example database work so well is SMTPit's ability to retain settings from one email to the next. Generally, the content of mass mail-outs is the exact same for every email. So, to be more efficient, SMTPit allows you set the Body of the email once at the very beginning, and then use that same Body for every email you send. You can do the same with any of the other SMTPit settings. The Mass Mailer Example database also keeps the same From Email Address and Subject for all emails.

This example database also makes use of the SMTP-Connect and SMTP-Disconnect functions in addition to the SMTP-Send function. The SMTP-Send function connects you to the mail server, sends a single email, and then disconnects, if you use it by itself, however, this is not very efficient in a mass mail-out solution. So, instead of connecting and disconnecting for every single email, you can use the SMTP-Connect and SMTP-Disconnect functions before and after your mass mail-out loop. If you call SMTP-Connect, then SMTP-Send will see that it's already connected, and just simply send the email. This is much more efficient. Please note, though, that if you do use SMTP-Connect, you must use SMTP-Disconnect when you are done. Otherwise, SMTPit will never disconnect you from your mail server.

Go Script

The following is an explaination of the Go script:

        Enter Browse Mode []
        Go to Layout ["Main"]

These two Script Steps just make sure you are in the right mode and on the right layout.

        Replace Contents [No Dialog, "Send Result", """"]

This line uses the Replace Contents Script Step to run through every record and clear out the Send Result field. If you need to stop your mass mail-out for one reason or another, then you will also want to be able to find where you left off in the database. The easiest way to do this is to look and see which is the first record that does not have something in the Send Result field. However, if you have run this script before, then the Send Result fields of all of the records would have the results from the last send, making it impossible to find where it left off. This is why all the Send Result fields are cleared out with a Replace Contents Script Step before we start a new mail-out.

        Set Field ["Start Time", "Status(CurrentTime)"]

This sets the Start Time field with the current time so that when we are done, we can see how long it took to send all of the emails.

        Perform Script [Sub-scripts, " _ Connect"]

This line calls our Connect script to set up the email and connect to the mail server. This script contains the following calculation:

```
Set Field ["Setup Result",
        External("SMTP-ClearAll", "") & "¶" &
        External("SMTP-HostAssign", SMTP Host) & "¶" &
        External("SMTP-AuthTypeAssign", Authentication Type) & "¶" &
        External("SMTP-UsernameAssign", Username) & "¶" &
        External("SMTP-PasswordAssign", Password) & "¶" &
        External("SMTP-FromAssign", From Email Address) & "¶" &
        External("SMTP-SubjectAssign", Subject) & "¶" &
        External("SMTP-BodyAssign", Text Body) & "¶" &
        External("SMTP-HTMLBodyAssign", HTML Body) & "¶" &
        External("SMTP-Connect", "")]
```

Most of this should look familiar from the Quick Start Example and Simple Mailer Example databases. There are two differences, however, that should be pointed out. First, there is no SMTP-ToAssign in this calculation. The reason for this is that this is the setup script that happens before our main mail-out loop. The To Email Addresses will be different for each record, so we want to use the To Email Address inside our main mail-out loop, not before it. The second difference is the new SMTP-Connect function on the last line. This connects you to your mail server so that you can start sending all of your emails.

```
If ["PatternCount(Setup Result, "ERROR:") > 0"]
    Show Message ["There was an error setting up the email. Please check the Setup
        Result field to determine the cause."]
```

Next we go back to our Go script and look at the next line. This line uses the If Script Step to see if anything in the Connect script had problems. If it can find the word "ERROR:" in the Setup Result field, then something did not work right, and it tells the user there was a problem with a Show Message Script Step.

```
Go to Layout [Refresh window, "List"]
Go to Record/Request/Page [First]
```

If, however, the Connect script was successful, then we go to the Go To Layout Script Step, which takes us to the List View layout so we can watch as SMTPit sends each of the emails. The next line will take us to the top of this list with the Go to Record/Request/Page Script Step with the "First" option. Using this Script Step does not affect the Sort Order, so if you have several contacts and you want to send them in a certain order, like alphabetically by Contact Name, then you can sort your records first, and then run the Go script.

```
Loop
    Perform Script [Sub-scripts, " \_ Send One"]
```

These Script Steps contain our main mail-out loop. This is the section of the script that gets repeated over and over for every single one of the records in the database. The Perform Script Step calls the Send One script which sends an email with the Contact information in the current record. We will look at this script in a second.

Flush Cache to Disk

This uses a lesser known Script Step called Flush Cache to Disk. To be more efficient and responsive to the user, FileMaker Pro keeps all database changes in your computer's RAM and only writes those database changes to disk when the program is idle, like when you are reading a field in the database. This is normal operation and great for when you are doing your normal daily database operations, however it is not so great for a mass mail-out solution.

The reason it is not so great is that FileMaker Pro never has any idle time to write those changes to disk when you are running through 1000 or so records sending emails to each one. If FileMaker Pro never has any idle time to write those changes to disk, then the changes start stacking up in RAM. This can be a bad thing for a number of reasons. Your computer or FileMaker Pro could crash for some unknown reason and lose all of those database changes since they were never written to disk. If FileMaker Pro's memory gets full of these database changes that have not been written to disk, and you try to make another change, then FileMaker Pro will finally give in and clear out it's memory by writing all the changes to disk. This can make it look like FileMaker Pro has stopped responding and can take a long time if you have allocated a lot of RAM to FileMaker Pro.

So, to remedy this situation, you need some way to tell FileMaker Pro to write all of the database changes to disk when you want. This is the purpose of the Flush Cache to Disk Script Step. This tells FileMaker Pro to write any and all database changes to your hard drive, immediately.

Go to Record/Request/Page [Exit after last, Next]

After you have sent the email for the current record, and flushed those database changes to disk, then you need to go to the next record. This line uses the Go to Record/Request/Page Script Step, this time with the "Next; Exit after Last" option to do this. The extra "Exit after Last" option allows you to break out of the main mail-out loop when you have come to the end of your list of contacts.

Refresh Window []

This Script Step simply refreshes or redraws the screen so that you can watch it progress through your records. Without this Script Step, you would just see a blank screen after the first few records have sent, and the only way you could tell it was working is it would flicker slightly every once in a while.

Go to Layout [Refresh window, "Main"]

This takes you back to the Main Layout after it has completed sending all of the emails.

Perform Script [Sub-scripts, " _ Disconnect"]

This Script Step calls the Disconnect script which simply uses the [SMTP-Disconnect](#) function to disconnect from your mail server.

```
Set Field ["End Time", "Status(CurrentTime)"]
Perform Script [Sub-scripts, " \_ Calculate Duration"]
```

The Set Field Script Step sets the End Time field with the current time, and the last line calls the Calculate Duration script which fills in the number of hours, minutes, and seconds it took to send your complete mail-out.

Now let us take a look back at the Send One script called from inside our loop.

```
If ["not IsEmpty(Contact Email Address)"]
```

The first line looks to see if this record has an email address. If it does not, then this script will exit. If the record does have an email address, then it goes to the next Script Step.

```
If ["Contact Name <> """]
```

This line determines if the current record has a Contact Name.

```
External("SMTP-ToAssign", """" & Contact Name & """ <" & Contact Email Address &
        ">")
```

If there is a Contact Name, this line is run. This calculation simply takes the Contact Name and Contact Email Address and makes it look like "Doe, John" <John@somewhere.com>.

```
External("SMTP-ToAssign", "<" & Contact Email Address & ">")
```

If there is no Contact Name, then all we have is the Contact Email Address. This line uses this calculation, which simply turns the Contact Email Address into <John@somewhere.com>.

```
Set Field ["Send Result",
        Send Result & "¶" &
        External("SMTP-EmailHeaderAppend", "X-RecID=" & NumToText(Contact
                ID)) & "¶" &
        External("SMTP-HeaderAssign", Text Header) & "¶" &
        External("SMTP-HTMLHeaderAssign", HTML Header) & "¶" &
        External("SMTP-FooterAssign", "---------------¶" & "You received this email be
                cause you showed interest in our product. If you no longer wish to receive
                emails from us, please send email to """ & From Email Address & """ with
                the subject ""REMOVE " & Contact Email Address & """. We are sorry for
                any inconvenience this email has caused you. ...") & "¶" &
        External("SMTP-HTMLFooterAssign", "<p><hr></p>¶" & "<p>You received this
                email be cause you showed interest in our product. If you no longer wish
                to receive emails from us, please <a href=""mailto:" & From Email
                Address & "?subject=REMOVE " & Contact Email Address & """>click
                here</a>. We are sorry for any inconvenience this email has caused you.
                ...")]
```

This Set Field Script Step contains the rest of the setup for this single record. This calculation is using several External Functions to enter in all sorts of information about the current record. It first uses the unique SMTP-EmailHeaderAppend function to add an email header to the header portion of the email. (The header portion of an email is the portion that contains the To, From, Subject, and other similar lines.) We use the SMTP-EmailHeaderAppend function to add a header that looks like "X-RecID: 47". In every mass mail-out you do, there will most likely be email addresses that were invalid or no longer in use. For those bad email addresses, you will receive a bounced or returned email saying the email could not be delivered. The majority of these bounced emails will at least have the header portion of the original email. So, by putting the "X-RecID" email header in the email, you can quickly take a look at the bounced message, find the "X-RecID" email header, and know exactly which record to look at in your database to correct or delete. You could also use an email receiving plug-in, like our popular POP3it plug-in, to receive these bounced messages and remove or flag bad email addresses automatically.

The SMTP-HeaderAssign and SMTP-HTMLHeaderAssign functions insert our special "Dear xxx," greetings at the top of the Body of our email. Take note that the header portion of an email is not the same as the header section of the body of the email. The header portion of an email contains all the To, From, Subject and other similar lines, while the header section of the body of the email is an optional section that goes before the Body of your email much like the footer section goes after the Body of your email.

The SMTP-FooterAssign and SMTP-HTMLFooterAssign functions insert special remove instructions for the recipient of the message to follow if they no longer wish to receive emails from you. The versions here in this documentation are much shorter than the version in the actual Mass Mailer Example database for simplicity. If you want to get rid of the footer message, you can delete these two functions or change them to fit your needs. These two lines show you how you can build part of the message "on the fly" with information in your record.

That concludes the Mass Mailer Example database. Hopefully you now have a better understanding of how to use SMTPit in a mass mail-out solution. Feel free to use this database as an example to build your own, or use this as a starting point for your own database.

HTML Image Example Explained

The HTML Image Example database is our final example database for SMTPit. The HTML Image feature allows you to include inline graphics for use in your HTML emails. Normally, if you want to have images in your HTML emails, you have to place those images on a publicly available website. Then when the recipient of your email views your HTML email, it can download the images from that website and display them to the user. Unfortunately, there are many drawbacks to this. The recipient of your email could be viewing your email "offline" or while they are not connected to the internet. If they are not connected to the internet, then they cannot download the image to view it. It could also be quite slow for the recipient's email client to contact the website and download the image.

Having inline HTML images fixes this problem by including the actual image in the email, sort of like a file attachment. You can then set up your HTML img tags to use the image embedded in the email instead of downloading it from a website. Be warned, though, that you should not go overboard with this and include a lot of graphics or include large graphics. Just like it can take a long time to download an image from a website, it can also take a long time to download an email that has several graphics or large graphics in it. You should try and keep the images small and use no more than 3 or 4, otherwise your recipients may not like it, especially if they have a slow connection to the internet.

With that in mind, let us look at the HTML Image Example database. You should find that the database contains the familiar fields from the other databases including To, From, Subject, Body, and Result. A new field is here, though, called HTML Images. In this field, you enter the Full Paths and FileNames of the HTML Images that you want to include in your email. Separate multiple images with a return. You can also use the Add button to use a standard open file dialog to select images to add to your email.

Also take a look at the HTML code in the HTML Body field. You should see a couple HTML img tags in the body. They look something like this:

        <img src="cid:part1">

The "src" parameter of the img tag is the key to showing the correct images at the correct locations. The parameter must always contain "cid:part#" where "#" is the number of the attachment in the order that you assign them. When you use the [SMTP-HTMLImageAssign](#) and [SMTP-HTMLImageAppend](#) functions, you need to remember the order in which you add the images. Let us say you have two graphic files, named "logo.gif" and "product.jpg", and you add them in that order, with "logo.gif" being the first, and "product.jpg" being the second. Then when you need to insert the "logo.gif" file in your email, you would specify the src of the img tag as "cid:part1", and when you need to insert the "product.jpg" file in your email, you would specify the src of the img tag as "cid:part2".

There are three extra things to note about HTML Images. First, if you are always going to be sending a specific graphic with every single one of your emails, then you can specify that graphic in the Default HTML Images in the Configuration Dialog. Any graphics you have specified there will always send with every email. They will also be placed first in the list of images. So, if you have two images in the Default HTML Images in the Configuration Dialog, and then you add two images with the [SMTP-HTMLImageAssign](#) function, the two images you assign with the

function will be the third and fourth images, which would correspond to "cid:part3" and "cid:part4".

The second thing to note is that if you do have multiple images they do not need to be sequential in your email. In other words, an img tag with the src "cid:part3" could come before an img tag with the src "cid:part1" in your HTML code. The final thing to note is if you want to use the same graphic in multiple places, you don't need to attach it multiple times. For example, if you had a list of items in your email and you want a tiny version of your logo to be in front of each of those line items, then you can use the same "cid:part2" img src for each of them, if your mini logo graphic was the second image assigned.

Once you fill in all the fields with your information, and insert a few image FileNames, you can press the Send button to send the email. This button calls the Send Message script which contains the following calculation:

```
Set Field ["Result",
        External("SMTP-ClearAll", "") & "¶" &
        External("SMTP-ToAssign", To) & "¶" &
        External("SMTP-FromAssign", From) & "¶" &
        External("SMTP-SubjectAssign", Subject) & "¶" &
        External("SMTP-HTMLBodyAssign", HTML Body) & "¶" &
        External("SMTP-HTMLImageAssign", HTML Images) & "¶" &
        External("SMTP-Send", "Transcript=yes")]
```

The first 5 functions should look familiar to you from the examples explained earlier in this documentation. SMTP-HTMLImageAssign is the new function that assigns the HTML Images you have defined in the HTML Images field to the email you are creating. Finally, the last line sends the email.

This concludes the explanations of the example databases. We hope that you now have a complete understanding of the possibilities of sending emails using SMTPit. Please keep in mind that the example databases we provide are just that, examples. They are by no means the only way to use SMTPit. We encourage you to play with the example databases, check out some of the other functions available in SMTPit, and to try out different things. If you have any questions about any of the example databases, and you cannot find answers in this documentation, then check our website dedicated to SMTPit at http://www.smtpit.com/. If you still cannot find answers to your questions, please feel free to email us at smtpit@cnsplug-ins.com.

SMTPit External Function Reference

Available Functions

The following is a list of all available functions and a complete description of each. For general information on how the External () functions work and how you can use them in your calculations, please see the How to use Plug-ins PDF Document distributed with this Document.

Functions with an asterisk (*) are new with version 3.1. Functions with a dagger (†) have changed with version 3.1.

SMTP-AssignAll

Use this function to assign any or all of the following settings at once: Host, To, From, CC, BCC, Subject, Port, AuthType, Username, and/or Password. The SMTP-AssignAll function will only set those settings; for all other SMTPit values, you must use their Assign and Append functions (see below). Note: The settings can be in any order, but you must separate them with the paragraph mark (¶) or with the pipe character (|). Also note: You do not have to use this function to set these settings, this is just one way to set them. You can use the other Assign and Append functions below, some of which have better support for their specific setting.

Example:

> External("SMTP-AssignAll", "Host=smtp.your.com¶To=theiremail@their.com¶From=youremail@your.com")
> External("SMTP-AssignAll", "CC=otherpeople@their.com¶BCC=blindcopies@their.com¶Subject=your subject")

See also:

> [SMTP-ClearAll](#)

SMTP-AttachAppend

The SMTP-AttachAppend function works exactly like the [SMTP-AttachAssign](#) function except that it adds attachments to your current list of attachments instead of overwriting them. You can append more than one file with this function by separating the file names with the paragraph mark (¶) or, if in a field, with a return.

Example:

> External("SMTP-AttachAppend", "c:\My Documents\resume.doc")
> External("SMTP-AttachAppend", "Macintosh HD:Documents:resume.doc")
> External("SMTP-AttachAppend", "c:\pdfs\Manual.pdf¶c:\pdfs\How to Install.pdf")

External("SMTP-AttachAppend", "Macintosh HD:PDFs:Manual.pdf¶Macintosh HD:PDFs:How to Install.pdf")

See also:

SMTP-AttachAssign
SMTP-DlgAttachAppend
SMTP-FileNameAcquire
SMTP-PathToDBAcquire
SMTP-PathToFMAcquire

SMTP-AttachAssign

The SMTP-AttachAssign function allows you to attach a file from your hard drive or network to the current email being sent. You must specify the full path and filename of the file. You can assign more than one file with this function by separating the file names with the paragraph mark (¶) or, if in a field, with a return. Note: If you use the SMTP-AttachAssign function twice, the second one will overwrite the first. If you have attachments assigned in the Configuration Dialog, they will always be sent with your email.

Example:

External("SMTP-AttachAssign", "c:\My Documents\resume.doc")
External("SMTP-AttachAssign", "Macintosh HD:Documents:resume.doc")
External("SMTP-AttachAssign", "c:\pdfs\Manual.pdf¶c:\pdfs\How to Install.pdf")
External("SMTP-AttachAssign", "Macintosh HD:PDFs:Manual.pdf¶Macintosh HD:PDFs:How to Install.pdf")
External("SMTP-AttachAssign", MyEmailAttachmentsField)

See also:

SMTP-AttachAppend
SMTP-DlgAttachAssign
SMTP-FileNameAcquire
SMTP-PathToDBAcquire
SMTP-PathToFMAcquire

SMTP-AuthTypeAssign

If your mail server requires authentication (requires you to log in) before it allows you to send email, then use the SMTP-AuthTypeAssign function to assign the type of authentication your mail server requires. The types of authentication SMTPit supports are (in increasing levels of security): None (off), Plain, Login, and CRAM-MD5.

Example:

>
> External("SMTP-AuthTypeAssign", "NONE")
> External("SMTP-AuthTypeAssign", "PLAIN")
> External("SMTP-AuthTypeAssign", "LOGIN")
> External("SMTP-AuthTypeAssign", "CRAM-MD5")

See also:

> SMTP-Connect
> SMTP-HostAssign
> SMTP-PasswordAssign
> SMTP-Send
> SMTP-UsernameAssign

SMTP-BCCAppend

The SMTP-BCCAppend function will add email addresses to the current list of "BCC" email addresses. If you have not set an email address with SMTP-BCCAssign, SMTP-BCCAppend will assign the current value to the "BCC" setting. You can use SMTP-BCCAppend multiple times without overwriting previous settings, unlike the SMTP-BCCAssign function. You can append multiple email addresses with this function. If you have multiple email addresses and you want to append them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

> External("SMTP-CCAppend", "smtpit@cnsplug-ins.com")
> External("SMTP-CCAppend", "<pop3it@cnsplug-ins.com>¶SMTPit Support <smtpit@cnsplug-ins.com>")

See also:

> SMTP-BCCAssign
> SMTP-CCAssign
> SMTP-CCAppend
> SMTP-FromAssign
> SMTP-ToAppend
> SMTP-ToAssign

SMTP-BCCAssign

Use the SMTP-BCCAssign function to assign the "BCC" (Blind Carbon Copy) email address(es). Note: If you use the SMTP-BCCAssign function twice, the second will overwrite the first. Any email addresses you have in the BCC list will have the email sent to them, but their email will not show up in the actual email. You can assign multiple email addresses with this function. If you have multiple email addresses and you want to set them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

> External("SMTP-BCCAssign", """Doe, John"" <John@somewhere.com>")
> External("SMTP-BCCAssign", "smtpit@cnsplug-ins.com¶info@cnsplug-ins.com")

See also:

> SMTP-BCCAppend
> SMTP-CCAssign
> SMTP-CCAppend
> SMTP-FromAssign
> SMTP-ToAssign
> SMTP-ToAppend

SMTP-BodyAppend

The SMTP-BodyAppend function will add to the Plain Text Body of the email assigned by SMTP-BodyAssign. If you have not set a body with SMTP-BodyAssign, SMTP-BodyAppend will assign the current value to the "Body" of the email. You can use SMTP-BodyAppend multiple times without overwriting previous settings, unlike the SMTP-BodyAssign function.

Example:

> External("SMTP-BodyAppend", portal_relation::Item Name & "   " & portal_relation::Item Cost)

See also:

> SMTP-BodyAssign
> SMTP-FileBodyAppend
> SMTP-FileBodyAssign

SMTP-FooterAppend
SMTP-FooterAssign
SMTP-HeaderAppend
SMTP-HeaderAssign

SMTP-BodyAssign

Use the SMTP-BodyAssign function to assign the Plain Text Body of the email. The "Body" of the email is the content of the email that the receiver will read. Note: If you use the SMTP-BodyAssign function twice, the second will overwrite the first.

Example:

External("SMTP-BodyAssign", MyEmailBodyField)
External("SMTP-BodyAssign", "Hi!")

See also:

SMTP-BodyAppend
SMTP-FileBodyAppend
SMTP-FileBodyAssign
SMTP-FooterAppend
SMTP-FooterAssign
SMTP-HeaderAppend
SMTP-HeaderAssign

SMTP-CCAppend

The SMTP-CCAppend function will add email addresses to the current list of "CC" email addresses. If you have not set an email address with SMTP-CCAssign, SMTP-CCAppend will assign the current value to the "CC" setting. You can use SMTP-CCAppend multiple times without overwriting previous settings, unlike the SMTP-CCAssign function. You can append multiple email addresses with this function. If you have multiple email addresses and you want to append them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

External("SMTP-CCAppend", "smtpit@cnsplug-ins.com")
External("SMTP-CCAppend", "<John@somewhere.com>¶SMTPit Support <smtpit@cnsplug-ins.com>")

See also:

> SMTP-BCCAppend
> SMTP-BCCAssign
> SMTP-CCAssign
> SMTP-FromAssign
> SMTP-ToAppend
> SMTP-ToAssign

SMTP-CCAssign

Use the SMTP-CCAssign function to assign the "CC" (Carbon Copy) email address(es). Note: If you use the SMTP-CCAssign function twice, the second will overwrite the first. You can assign multiple email addresses with this function. If you have multiple email addresses and you want to set them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

> External("SMTP-CCAssign", """Doe, John"" <John@somewhere.com>")
> External("SMTP-CCAssign", "smtpit@cnsplug-ins.com¶info@cnsplug-ins.com")

See also:

> SMTP-BCCAppend
> SMTP-BCCAssign
> SMTP-CCAppend
> SMTP-FromAssign
> SMTP-ToAppend
> SMTP-ToAssign

SMTP-ClearAll

The SMTP-ClearAll function clears all settings that have been set with the External Functions so that the next email can start with a clean slate. The default settings in the Configuration Dialog are not cleared and will be used in your next email unless you set new values with the External Functions. You can also set SMTPit to automatically clear or not clear specific fields on the "After Send" tab in the Configuration Dialog. The parameter should be the empty string ("").

Example:

       External("SMTP-ClearAll", "")

See also:

       SMTP-AssignAll

SMTP-Connect

This function allows you to manually connect to your mail server instead of letting the SMTP-Send function connect for you. When you use the SMTP-Send function by itself, it will connect to your mail server, send your email, and then disconnect. If you call SMTP-Connect before calling SMTP-Send, then SMTP-Send will only send your email; it will not try to connect or disconnect. This is useful in mass-email solutions that need to send out several emails at a time. Connecting once, sending several emails, and then disconnecting is much more efficient than connecting and disconnecting for each individual email. If you call SMTP-Connect, you must call SMTP-Disconnect, otherwise SMTPit will never disconnect you from the mail server. In a mass-email solution, you would want to call SMTP-Connect before your looping script, and call SMTP-Disconnect after your looping script. For normal operation, use the empty string as the parameter ("").

The SMTP-Connect function can take optional arguments. If you would like a transcript of the interaction between SMTPit and your mail server, use "Transcript=Yes". This may be useful when determining why your mail server is not allowing you to connect to it. If you want to specify more than one argument, separate them with a semicolon.

If your mail server requires authentication before you can send email, you must set up the Authentication Type, Username, and Password before calling this function. Otherwise, SMTPit will not have the required information to properly authenticate with your mail server.

Example:

       External("SMTP-Connect", "")
       External("SMTP-Connect", "Transcript=Yes")

See also:

       SMTP-AuthTypeAssign
       SMTP-Disconnect
       SMTP-HostAssign
       SMTP-PasswordAssign
       SMTP-PortAssign
       SMTP-Send
       SMTP-UsernameAssign

SMTP-Disconnect

If you use SMTP-Connect to connect to your mail server manually (instead of letting SMTP-Send do it), then you must call SMTP-Disconnect to disconnect from your mail server. Otherwise, SMTPit will never disconnect you from the mail server. The parameter should be the empty string ("").

Example:

External("SMTP-Disconnect", "")

See also:

SMTP-Connect
SMTP-Send

SMTP-DlgAttachAppend

The SMTP-DlgAttachAppend function works exactly like the SMTP-DlgAttachAssign function except that it adds attachments to your current list of attachments. The parameter should be the empty string ("").

Example:

External("SMTP-DlgAttachAppend", "")

See also:

SMTP-AttachAppend
SMTP-DlgAttachAssign

SMTP-DlgAttachAssign

This function presents you with a standard open file dialog that lets you choose a file from your hard drive. It then attaches the chosen file to the current email. The parameter should be the empty string ("").

Example:

External("SMTP-DlgAttachAssign", "")

See also:

> SMTP-AttachAssign
> SMTP-DlgAttachAppend

SMTP-DlgHTMLImageAppend

The SMTP-DlgHTMLImageAppend function is similar to the SMTP-DlgAttachAppend function. This function presents you with a standard open file dialog that lets you choose a graphic image file on your hard drive. It then adds it to your current list of HTML Images. For more information on inline HTML images, see "HTML Image Example Explained" earlier in this documentation. The parameter should be the empty string ("").

Example:

> External("SMTP-DlgHTMLImageAppend", "")

See also:

> SMTP-DlgHTMLImageAssign
> SMTP-HTMLImageAppend

SMTP-DlgHTMLImageAssign

The SMTP-DlgHTMLImageAssign function is similar to the SMTP-DlgAttachAssign function. This function presents you with a standard open file dialog that lets you choose a graphic image file from your hard drive. It then uses the chosen graphic as an inline graphic image for you to use in your HTML emails. For more information on inline HTML images, see "HTML Image Example Explained" earlier in this documentation. The parameter should be the empty string ("").

Example:

> External("SMTP-DlgHTMLImageAssign", "")

See also:

> SMTP-DlgHTMLImageAppend
> SMTP-HTMLImageAssign

SMTP-EmailHeaderAppend

The SMTP-EmailHeaderAppend function adds extra email headers to your email messages. Note that there should be no spaces in the name section of the header, however the value can have spaces. This function looks like the following:

> External("SMTP-EmailHeaderAppend", "name=email header value")

The first part is the name and the second part is the value corresponding to the name part. Use an equal sign to separate them. Note: You can also set Default Email Headers in the Configuration Dialog. If you have Email Headers assigned in the Configuration Dialog, they will be appended to the Header of the email before any Email Headers you set with this function. You cannot set any of the following with this function: To, From, Subject, Date, Message-ID, CC, or BCC.

You can assign multiple email headers with this function. If you have multiple email headers and you want to set them all at once, separate each email header with a paragraph mark (¶) or, if in a field, with a return.

Example:

> External("SMTP-EmailHeaderAppend", "X-Company=My Company")
> External("SMTP-EmailHeaderAppend", "X-RecID=" & RecordID)
> External("SMTP-EmailHeaderAppend", "X-Company=" & Company Name & "¶X-URL=" & Company URL)
> External("SMTP-EmailHeaderAppend", MyEmailHeadersField)

SMTP-FileBodyAppend

The SMTP-FileBodyAppend function will add the contents of a text file to the Plain Text Body of the email. You can use this function several times to add different text files to the Plain Text Body without overwriting any previous settings.

Example:

> External("SMTP-FileBodyAppend", "c:\My Documents\email signature.txt")
> External("SMTP-FileBodyAppend", "Macintosh HD:Documents:email signature.txt")

See also:

> SMTP-BodyAppend
> SMTP-BodyAssign
> SMTP-FileBodyAssign
> SMTP-FooterAppend
> SMTP-FooterAssign

SMTP-HeaderAppend
SMTP-HeaderAssign

SMTP-FileBodyAssign

You can use the SMTP-FileBodyAssign function to load a text file into the Plain Text Body of the email. Note: If you use SMTP-BodyAssign or SMTP-FileBodyAssign a second time, it will overwrite the previous setting.

Example:

External("SMTP-FileBodyAssign", "c:\My Documents\email body.txt")
External("SMTP-FileBodyAssign", "Macintosh HD:Documents:email body.txt")

See also:

SMTP-BodyAppend
SMTP-BodyAssign
SMTP-FileBodyAppend
SMTP-FooterAppend
SMTP-FooterAssign
SMTP-HeaderAppend
SMTP-HeaderAssign

SMTP-FileNameAcquire

This function allows you to use a standard open file dialog to choose a file, and then returns the full path and filename of the selected file. Note: This function only returns a path to a file to you. It does nothing with the actual file. It does not use the selected file as an attachment or HTML image. The parameter should be the empty string ("").

Example:

External("SMTP-FileNameAcquire", "")

See also:

SMTP-AttachAppend
SMTP-AttachAssign
SMTP-HTMLImageAppend
SMTP-HTMLImageAssign
SMTP-PathToDBAcquire
SMTP-PathToFMAcquire

SMTP-FolderAcquire*

This function allows you to use a standard open file dialog to choose a folder, and then returns the full path of the selected folder. Note: This function only returns a path to a folder to you. It does nothing with the actual folder. The parameter should be the empty string ("").

Example:

       External("SMTP-FolderAcquire", "")

See also:

       SMTP-FileNameAcquire
       SMTP-PathToDBAcquire
       SMTP-PathToFMAcquire

SMTP-FooterAppend

Like the SMTP-HeaderAppend, the SMTP-FooterAppend appends text to the current Plain Text Body Footer. If you have not previously used the SMTP-FooterAssign function, but you have a Default Footer defined in the Configuration Dialog, this function will append to the Default Footer.

Example:

       External("SMTP-FooterAppend", Company Disclaimer)

See also:

       SMTP-BodyAppend
       SMTP-BodyAssign
       SMTP-FooterAssign
       SMTP-HeaderAppend
       SMTP-HeaderAssign

SMTP-FooterAssign

This is another unique function in SMTPit that will allow you to have a "Footer" section in the Plain Text Body of your email. Possible uses include an email signature, a public key, an email advertisement, quotes, or a passage telling how to unsubscribe from your list. You can set a Default Footer in the Configuration Dialog. If you assign a Plain Text Footer using SMTP-FooterAssign, SMTPit will ignore the default footer.

Example:

> External("SMTP-FooterAssign", "-- ¶John Doe¶John@somewhere.com")

See also:

> SMTP-BodyAppend
> SMTP-BodyAssign
> SMTP-FooterAppend
> SMTP-HeaderAppend
> SMTP-HeaderAssign

SMTP-FromAssign

Use the SMTP-FromAssign function to assign the "From" email address. Note: If you use the SMTP-FromAssign function twice, the second one will overwrite the first. If you assign a from email address using this function, SMTPit will ignore the Default From field in the Configuration Dialog. SMTPit will validate the email address you give it to see if it "looks" like an email address. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

> External("SMTP-FromAssign", "smtpit@cnsplug-ins.com")
> External("SMTP-FromAssign", "SMTPit Support <smtpit@cnsplug-ins.com>")
> External("SMTP-FromAssign", Company Name & "Support <" & Company Email & ">")

See also:

> SMTP-BCCAppend
> SMTP-BCCAssign
> SMTP-CCAppend
> SMTP-CCAssign
> SMTP-ToAppend
> SMTP-ToAssign

SMTP-HeaderAppend

The SMTP-HeaderAppend will append text to the current Plain Text Body Header. If you have not previously used the SMTP-HeaderAssign function, but you have a Default Header defined in the Configuration Dialog, this function will append to the Default Header.

Example:

       External("SMTP-HeaderAppend", "------------------¶¶")

See also:

       SMTP-BodyAppend
       SMTP-BodyAssign
       SMTP-FooterAppend
       SMTP-FooterAssign
       SMTP-HeaderAssign

SMTP-HeaderAssign

This is a unique function that will allow you to have a "Header" section in the Plain Text Body of your email message. Possible uses are email letterheads or personalized email messages for mass emailing. SMTPit also allows you to set a Default Header in the Configuration Dialog. If you assign a Plain Text Header using SMTP-HeaderAssign, SMTPit will ignore the Default Header.

Example:

       External("SMTP-HeaderAssign", "Dear " & Contact Name & ",¶¶")

See also:

       SMTP-BodyAppend
       SMTP-BodyAssign
       SMTP-FooterAppend
       SMTP-FooterAssign
       SMTP-HeaderAppend

SMTP-HideStatus

Use this function to hide the status window that you had previously shown with the SMTP-ShowStatus function. You must call this function to hide the status window before closing the FileMaker Pro application (see SMTP-ShowStatus). Power-users will see that this function returns the last known coordinates of the status window, which can be extracted and stored so that the next time the status window is shown, it can be shown in the same place. The parameter should be the empty string ("").

Example:

> External("SMTP-HideStatus", "")

See also:

> SMTP-MoveStatus
> SMTP-ShowStatus

SMTP-HostAssign

The host is the domain name or IP address of the SMTP mail server you are connecting to. You can assign the host in your scripts, or set up a Default Host in the Configuration Dialog. SMTP-Send and SMTP-Connect will return an error if no host has been set. If you assign a host with this function, SMTPit will ignore the Default Host in the Configuration Dialog.

Example:

> External("SMTP-HostAssign", "mail.yourdomain.com")

See also:

> SMTP-AuthTypeAssign
> SMTP-Connect
> SMTP-PasswordAssign
> SMTP-PortAssign
> SMTP-Send
> SMTP-UsernameAssign

SMTP-HTMLBodyAppend

This function works exactly like the SMTP-BodyAppend function, except that it appends to the HTML Body of the email instead of the Plain Text Body of the email. You can use this function multiple times to append more HTML text.

Example:

> External("SMTP-HTMLBodyAppend", "<tr><td>portal_relation::Item Name & "</td><td>" & portal_relation::
> Item Cost & "</td></tr>")

See also:

>   SMTP-HTMLBodyAssign
>   SMTP-HTMLFileBodyAppend
>   SMTP-HTMLFileBodyAssign
>   SMTP-HTMLFooterAppend
>   SMTP-HTMLFooterAssign
>   SMTP-HTMLHeaderAppend
>   SMTP-HTMLHeaderAssign

SMTP-HTMLBodyAssign

This function works exactly like the SMTP-BodyAssign function, except that it sets the HTML Body of the email instead of the Plain Text Body of the email. Note: If you use this function twice, the second will overwrite the first.

Example:

>   External("SMTP-HTMLBodyAssign", MyHTMLEmailBodyField)
>   External("SMTP-HTMLBodyAssign", "<h1>Hi!</h1>")

See also:

>   SMTP-HTMLBodyAppend
>   SMTP-HTMLFileBodyAppend
>   SMTP-HTMLFileBodyAssign
>   SMTP-HTMLFooterAppend
>   SMTP-HTMLFooterAssign
>   SMTP-HTMLHeaderAppend
>   SMTP-HTMLHeaderAssign

SMTP-HTMLFileBodyAppend

This function works exactly like the SMTP-FileBodyAppend function, except that it appends to the HTML Body of the email instead of the Plain Text Body of the email. You can use this function several times to add different HTML files to the HTML Body without overwriting any previous settings.

Example:

>   External("SMTP-HTMLFileBodyAppend", "c:\My Documents\disclaimer.htm")
>   External("SMTP-HTMLFileBodyAppend", "Macintosh HD:Documents:disclaimer.htm")

See also:

> SMTP-HTMLBodyAppend
> SMTP-HTMLBodyAssign
> SMTP-HTMLFileBodyAssign
> SMTP-HTMLFooterAppend
> SMTP-HTMLFooterAssign
> SMTP-HTMLHeaderAppend
> SMTP-HTMLHeaderAssign

SMTP-HTMLFileBodyAssign

This function works exactly like the SMTP-FileBodyAssign function, except that it sets the HTML Body of the email instead of the Plain Text Body of the email. Note: If you use SMTP-HTMLBodyAssign, or SMTP-HTMLFileBodyAssign a second time, it will overwrite the previous setting.

Example:

> External("SMTP-HTMLFileBodyAssign", "c:\My Documents\email body.htm")
> External("SMTP-HTMLFileBodyAssign", "Macintosh HD:Documents:email body.htm")

See also:

> SMTP-HTMLBodyAppend
> SMTP-HTMLBodyAssign
> SMTP-HTMLFileBodyAppend
> SMTP-HTMLFooterAppend
> SMTP-HTMLFooterAssign
> SMTP-HTMLHeaderAppend
> SMTP-HTMLHeaderAssign

SMTP-HTMLFooterAppend

This function works exactly like the SMTP-FooterAppend function, except that it appends to the HTML Body Footer instead of the Plain Text Body Footer. If you have not previously used the SMTP-HTMLFooterAssign function, but you have a Default HTML Footer in the Configuration Dialog, this function will append to the Default HTML Footer.

Example:

> External("SMTP-HTMLFooterAppend", "<p>Visit my website!</p>")

See also:

> SMTP-HTMLBodyAppend
> SMTP-HTMLBodyAssign
> SMTP-HTMLFooterAssign
> SMTP-HTMLHeaderAppend
> SMTP-HTMLHeaderAssign

SMTP-HTMLFooterAssign

This function works exactly like the SMTP-FooterAssign function, except that it sets the HTML Body Footer instead of the Plain Text Body Footer. You can also assign a Default HTML Footer in the Configuration Dialog. If you assign an HTML Footer using the SMTP-HTMLFooterAssign, SMTPit will ignore the Default HTML Footer.

Example:

> External("SMTP-HTMLFooterAssign", "<p><hr>
John Doe
John@somewhere.com</p>")

See also:

> SMTP-HTMLBodyAppend
> SMTP-HTMLBodyAssign
> SMTP-HTMLFooterAppend
> SMTP-HTMLHeaderAppend
> SMTP-HTMLHeaderAssign

SMTP-HTMLHeaderAppend

This function works exactly like the SMTP-HeaderAppend function, except that it appends to the HTML Body Header instead of the Plain Text Body Header. If you have not previously used the SMTP-HTMLHeaderAssign function, but you have a Default HTML Header in the Configuration Dialog, this function will append to the Default HTML Header.

Example:

> External("SMTP-HTMLHeaderAppend", "<p><hr></p>")

See also:

> SMTP-HTMLBodyAppend
> SMTP-HTMLBodyAssign

SMTP-HTMLFooterAppend
SMTP-HTMLFooterAssign
SMTP-HTMLHeaderAssign

SMTP-HTMLHeaderAssign

This function works exactly like the SMTP-HeaderAssign function, except that it sets the HTML Body Header instead of the Plain Text Body Header. You can also assign a Default HTML Header in the Configuration Dialog. If you assign an HTML Header using SMTP-HTMLHeaderAssign, SMTPit will ignore the Default HTML Header.

Example:

External("SMTP-HTMLHeaderAssign", "<p>Dear " & Contact Name & ",</p>")

See also:

SMTP-HTMLBodyAppend
SMTP-HTMLBodyAssign
SMTP-HTMLFooterAppend
SMTP-HTMLFooterAssign
SMTP-HTMLHeaderAppend

SMTP-HTMLImageAppend

The SMTP-HTMLImageAppend function is similar to the SMTP-AttachAppend function. This function will add more graphic images files to your current list of HTML Images. You can append more than one file with this function by separating the file names with the paragraph mark (¶) or, if in a field, with a return. For more information on inline HTML images, see "HTML Image Example Explained" earlier in this documentation.

Example:

External("SMTP-HTMLImageAppend", "c:\Images\logo.gif")
External("SMTP-HTMLImageAppend", "Macintosh HD:Images:logo.gif")
External("SMTP-HTMLImageAppend", "c:\thumbnails\product1.jpg¶c:\thumbnails\product2.jpg")
External("SMTP-HTMLImageAppend", "Macintosh HD:Thumbnails:product1.jpg¶Macintosh HD: Thumbnails:product2.jpg")

See also:

SMTP-DlgHTMLImageAppend
SMTP-FileNameAcquire

SMTP-HTMLImageAssign
SMTP-PathToDBAcquire
SMTP-PathToFMAcquire

SMTP-HTMLImageAssign

The SMTP-HTMLImageAssign function is similar to the SMTP-AttachAssign function. This function allows you
to attach a graphic image file from your hard drive or network to the current email being sent. SMTPit takes this
graphic and makes it an inline graphic image for you to use in your HTML emails. You must specify the full path and
filename of the image. You can assign more than one file with this function by separating the file names with the
paragraph mark (¶) or, if in a field, with a return. If you have HTML Images assigned in the Configuration Dialog,
they will always be sent with your email, and they will be added to the email first. For more information on inline
HTML images, see "HTML Image Example Explained" earlier in this documentation. If you do not assign an HTML
Body, then an HTML email will not be sent and these HTML Images will be ignored.

Example:

External("SMTP-HTMLImageAssign", "c:\Images\logo.gif")
External("SMTP-HTMLImageAssign", "Macintosh HD:Images:logo.gif")
External("SMTP-HTMLImageAssign", "c:\thumbnails\product1.jpg¶c:\thumbnails\product2.jpg")
External("SMTP-HTMLImageAssign", "Macintosh HD:Thumbnails:product1.jpg¶Macintosh HD:Thumbnails:
product2.jpg")
External("SMTP-HTMLImageAssign", MyHTMLImagesField)

See also:

SMTP-DlgHTMLImageAssign
SMTP-FileNameAcquire
SMTP-HTMLImageAppend
SMTP-PathToDBAcquire
SMTP-PathToFMAcquire

SMTP-MoveStatus

Use this function to move the status window to a different location on the screen. The parameter is the same as the
parameter defined in the SMTP-ShowStatus function. Power-users will see that this function returns the coordinates
before and after the move, which can be extracted and stored so that the status window can be moved back to it's
previous location if desired.

Example:

> External("SMTP-MoveStatus", "700,100")
> External("SMTP-MoveStatus", "-1,600")
> External("SMTP-MoveStatus", "-1,-1")

See also:

> SMTP-HideStatus
> SMTP-ShowStatus

SMTP-PasswordAssign

If your mail server requires authentication (requires you to log in) before it allows you to send email, then use the SMTP-PasswordAssign function to assign the password of your email account. Note: You will also need to use the SMTP-AuthTypeAssign function to specify what type of authentication your mail server needs. Otherwise, SMTPit will ignore this setting.

Example:

> External("SMTP-PasswordAssign", "love")

See also:

> SMTP-Connect
> SMTP-HostAssign
> SMTP-Send
> SMTP-UsernameAssign

SMTP-PathToDBAcquire

If you need to store your attachments in the same folder as your database, either for convenience or in your bound solution, and you are not sure if that location will stay the same, then you need a way to find the location of your database on the hard drive. This function will return to you the path to the folder that contains your database. You can then add the name of your file to the end of this path for use with the Attachment and HTML Image functions. You must provide the name of the database as the parameter.

Example:

> External("SMTP-PathToDBAcquire", Status(CurrentFileName))

See also:

> SMTP-AttachAssign
> SMTP-AttachAppend
> SMTP-FolderAcquire
> SMTP-HTMLImageAssign
> SMTP-HTMLImageAppend
> SMTP-FileNameAcquire
> SMTP-PathToFMAcquire

SMTP-PathToFMAcquire*

This function returns the path to the folder that contains the running version of FileMaker. The parameter should be the empty string ("").

Example:

> External("SMTP-PathToFMAcquire", "")

See also:

> SMTP-AttachAppend
> SMTP-AttachAssign
> SMTP-FileNameAcquire
> SMTP-FolderAcquire
> SMTP-HTMLImageAppend
> SMTP-HTMLImageAssign
> SMTP-PathToDBAcquire

SMTP-PortAssign

Use this function to specify an alternate TCP/IP port for SMTPit to connect to on your mail server. SMTPit defaults to the standard SMTP TCP/IP port: 25. You will most likely never need to use this function.

Example:

> External("SMTP-PortAssign", "16384")

See also:

> SMTP-HostAssign

SMTP-PriorityAssign

This function allows you to set the "Priority" of an email. Most email clients understand the Priority setting on an email and will highlight the email as more or less important than other emails. This function will take the numbers 1 through 5 with 1 being the Highest Priority, and 5 being the Lowest. This function will also take one of the following five words: "Highest", "High", "Normal", "Low", or "Lowest". The Priority defaults to "Normal".

Example:

>   External("SMTP-PriorityAssign", "1")
>   External("SMTP-PriorityAssign", "High")
>   External("SMTP-PriorityAssign", Email Priority)

SMTP-Register †

This function allows you to register your copy of SMTPit via a function rather than using the Configuration Dialog. It is mostly meant for developers so that they can register plug-ins for bound solutions. For people who have Site Licenses that have a startup script that registers the plugin every time the solution starts, you can use the new "I Accept the License Agreement" parameter to automatically accept the new license agreement that pops up when the plugin is registered. The parameter consists of your first name, last name, serial number, and optional license agreement all separated by the pipe character ("|"). ("|" is created by typing shift-backslash.) You can also use "Dialog" as the parameter to make the register window open.

Examples:

>   External("doFTP-Register", "First Name|Last Name|Serial Number")
>   External("doFTP-Register", "First Name|Last Name|Serial Number|I Accept the License Agreement")
>   External("doFTP-Register", "Dialog")

SMTP-Send

The SMTP-Send function first checks to make sure you have assigned enough fields to send an email. If everything checks out, it attempts to send the email to your mail server. If there is a problem sending the email to your mail server, it will return the error. If the email sent successfully, it will return the string "Email Sent Successfully." For normal operation, use the empty string as the parameter ("").

The SMTP-Send function can take optional arguments. If you would like a dialog to pop up informing you that the email was sent, use "Dialog=Yes". This is the same as checking the option in the Configuration Dialog, but this allows you to do it on a per email basis. If you would like a transcript of the interaction between SMTPit and your mail server, use "Transcript=Yes". This may be useful when determining why your mail server is not allowing you to send email through it. If you want to specify more than one argument, separate them with a comma.

You can use this function by itself, or with the SMTP-Connect and SMTP-Disconnect functions. If you use this function by itself, this function will connect to your mail server, send the email, and then disconnect. If you call SMTP-Connect before calling this function, this function will simply send the email; it will not try to connect and disconnect. This is useful in mass-email solutions that send out several emails in one setting. Connecting once, sending several emails, and then disconnecting is much more efficient than connecting and disconnecting for each individual email. If you call SMTP-Connect, you must call SMTP-Disconnect, otherwise SMTPit will never disconnect you from the mail server.

If your mail server requires authentication before you can send email, you must set up the Authentication Type, Username, and Password before calling this function. Otherwise, SMTPit will not have the required information to properly authenticate with your mail server.

Example:

> External("SMTP-Send", "")
> External("SMTP-Send", "Dialog=Yes; Transcript=Yes")
> External("SMTP-Send", "Dialog=" & YesNoField)

See also:

> SMTP-AuthTypeAssign
> SMTP-Connect
> SMTP-Disconnect
> SMTP-HostAssign
> SMTP-PasswordAssign
> SMTP-PortAssign
> SMTP-UsernameAssign

SMTP-ShowStatus

Use this function to show a status window that displays information about what the plug-in is currently doing. When you are sending an email, the status window will show you what part of the email it is currently sending as well as a progress bar indicating how much of that part it has completed. If you specify an empty string ("") as the parameter, the status window will show up in the middle of the screen.

If you want to specify a starting location for the status window, specify the coordinates of the left and top of the dialog in pixels in the form "x,y" or "across,down". For instance, if you wanted to display it in the top right hand corner of your screen, and your screen resolution is set to 800x600, you could specify "700,100". If you specify a negative one ("-1") as either the x (across) or y (down) coordinates, the status window will be centered on that axis. For instance, if you want to display it on the bottom of your screen in the center, you would specify "-1,600"; or in the center of the screen by specifying "-1,-1".

On windows, this status window will always be on top of every other window and should never be hidden by another window. On macintosh, the status window can be hidden behind another open window. You can use the "Window" menu in the menu bar across the top of your screen to bring the status window to the front.

Also, on macintosh, you must ensure that the status window is closed before quitting the FileMaker Pro application. If the status window is still visible when you close the FileMaker Pro application, you will get an error message saying that FileMaker Pro cannot write to the disk, and give you the option to "Quit" or "Continue". You will only be able to "Quit", which is the equivalent of a force-quit, which does not safely free all its resources. The easiest way to ensure that the status window is closed when you close the FileMaker Pro application is to call SMTP-HideStatus in a "close" script that you define in Edit->Preferences->Document of your main database.

Example:

> External("SMTP-ShowStatus", "")
> External("SMTP-ShowStatus", "700,100")
> External("SMTP-ShowStatus", "-1,600")

See also:

> SMTP-HideStatus
> SMTP-MoveStatus

SMTP-SubjectAppend

The SMTP-SubjectAppend function will add to the current "Subject" of the email. If you have previously used the SMTP-SubjectAssign function, this function will add on to that setting. However, if you have not previously used the SMTP-SubjectAssignn function, but you have a value in the "Default Subject" field in the Configuration Dialog, this function will add to the default setting. You can use SMTP-SubjectAppend multiple times without overwriting previous assignments, unlike the SMTP-SubjectAssign function.

Example:

> External("SMTP-SubjectAppend", " Order Number: " & OrderID)

See also:

> SMTP-SubjectAssign

SMTP-SubjectAssign

Use the SMTP-SubjectAssign function to assign the "Subject" of the email. Note: If you use the SMTP-SubjectAssign function twice, the second will overwrite the first. If you assign a "Subject" of the email with this function, SMTPit will ignore the Default Subject field in the Configuration Dialog.
Example:

> External("SMTP-SubjectAssign", "In response to your inquiry")

See also:

> SMTP-SubjectAppend

SMTP-ToAppend

The SMTP-ToAppend function will add email addresses to the current list of "To" email addresses. If you have not set an email address with SMTP-ToAssign, SMTP-ToAppend will assign the current value to the "To" setting. You can use SMTP-ToAppend multiple times without overwriting previous settings, unlike the SMTP-ToAssign function. You can append multiple email addresses with this function. If you have multiple email addresses and you want to append them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error. For examples of what SMTPit considers to be a "valid" email address, see the SMTP-ToAssign function.

Example:

> External("SMTP-ToAppend", "<smtpit@cnsplug-ins.com>")
> External("SMTP-ToAppend", "<smtpit@cnsplug-ins.com> (SMTPit Support)¶John@somewhere.com")

See also:

> SMTP-BCCAppend
> SMTP-BCCAssign
> SMTP-CCAppend
> SMTP-CCAssign
> SMTP-FromAssign
> SMTP-ToAssign

SMTP-ToAssign

Use the SMTP-ToAssign function to assign the "To" email address(es). Once assigned you can add other email addresses by using the SMTP-ToAppend function. Note: If you use the SMTP-ToAssign function twice, the second will overwrite the first. You can assign multiple email addresses with this function. If you have multiple email addresses and you want to set them all at once, separate each email address with a paragraph mark (¶) or, if in a field, with a return. SMTPit will validate the email addresses you give it to see if they "look" like email addresses. If SMTPit cannot find a valid email address, it will return an error.
SMTPit considers all of the following to be "valid" email addresses:

smtpit@comm-unity.net
<smtpit@cnsplug-ins.com>
SMTPit Support <smtpit@cnsplug-ins.com>
"SMTPit Support" <smptit@cnsplug-ins.com>
"Doe, John" <John@somewhere.com>
<smtpit@cnsplug-ins.com> (SMTPit Support)

The basic rule is, if you have any "human readable" names in the email address (like "SMTPit Support"), then you need angled brackets (< and >) around the actual email address. This is so mail servers and mail clients can correctly identify the email address. If you have a comma in the "human readable" name, then you need to enclose it in double quotes ("") (like "Doe, John").

Example:

> External("SMTP-ToAssign", "SMTPit Support <smtpit@cnsplug-ins.com>")
> External("SMTP-ToAssign", "smtpit@cnsplug-ins.com¶""Doe, John"" <John@somewhere.com>")
> External("SMTP-ToAssign", MyEmailAddressesField)
> External("SMTP-ToAssign", """" & Contact Name & """ <" & Contact Email & ">")

See also:

> SMTP-BCCAppend
> SMTP-BCCAssign
> SMTP-CCAppend
> SMTP-CCAssign
> SMTP-FromAssign
> SMTP-ToAppend

SMTP-UsernameAssign

If your mail server requires authentication (requires you to log in) before it allows you to send email, then use the SMTP-UsernameAssign function to assign the username of your email account. Note: You will also need to use the SMTP-AuthTypeAssign function to specify what type of authentication your mail server needs. Otherwise, SMTPit will ignore this setting.

Example:

> External("SMTP-UsernameAssign", "frank")

See also:

> SMTP-Connect
> SMTP-HostAssign
> SMTP-PasswordAssign
> SMTP-Send

SMTP-Version

This function returns the current version of SMTPit when the parameter string is empty (""). You can also use this function to display the Configuration Dialog. If you pass it the parameter "CONFIGURE", the Configuration Dialog will open. If you pass it the parameter "ABOUT", it will display the Configuration Dialog with the "About" tab as the first tab shown.

Example:

> External("SMTP-Version", "")
> External("SMTP-Version", "CONFIGURE")
> External("SMTP-Version", "ABOUT")

Credits

Programming, documentation, and example databases by Jake Traynham
Concept, web design, and example databases by Jesse Traynham
Documentation, and example databases by Daniel Sims.

Contact Information

Email: SMTPit@cnsplug-ins.com
SMTPit Website: http://www.SMTPit.com/
Main Website: http://www.cnsplug-ins.com/
Phone: 817-560-4226
Fax: 817-244-0340

You can write us at:

Comm-Unity Networking Systems
8652 Camp Bowie West
Fort Worth, Texas 76116