Version 3.0

The Email Receiving Plug-in for FileMaker Pro 4.x / 5.x

Developed by Comm-Unity Networking Systems

Receiving email with
FileMaker Pro has never
been so easy and flexible!

Comm-Unity Networking Systems

POP3it Support Email:
pop3it@comm-unity.net

POP3it Support Website:
http://www.pop3it.com/

Table of Contents

Introduction to POP3it and FileMaker Pro Plug-ins

Introduction

POP3it is an exciting plug-in from Comm-Unity Networking Systems that allows you to receive email directly into your FileMaker Pro databases. With POP3it, you can check multiple POP3 email accounts, which brings incredible new power to FileMaker Pro.

The following documentation will help to familiarize yourself with the POP3it functions, the example databases included with the POP3it archive, and how to build your own POP3 email solutions. This plug-in is different than most FileMaker Pro plug-ins in the sense that it brings information into FileMaker Pro, whereas most plug-ins export information. Because of this difference, scripting POP3it is inherently different than other plug-ins.

Before we get to far into the documentation, let's take a look at the features of POP3it and some possible applications.

Features and Uses

POP3it has many features, including the following:

- Dynamically check multiple POP3 email accounts, even on different email servers.
- Automatically extract major email fields such as From, Subject, and Body.
- Receive attachments, which are placed in a directory that you can specify dynamically.
- Download entire messages into one field or split up into their respective fields.

POP3it has many potential uses. The following are a few ideas you could use POP3it in:

- Archive a mailing list in FileMaker Pro and use the Web Companion to build a web interface to that mailing list.
- Use SMTPit and POP3it together to create a complete email client using only FileMaker Pro.
- Create new records based on an email message.
- Synchronize two FileMaker Pro databases via email.

Installation and Configuration

To install the plug-in, first make sure FileMaker Pro is closed. Next, unzip the POP3it zip file on Windows, or unstuff the POP3it stuffit file on Macintosh, and then double-click the POP3it_Installer application. This will automatically place the POP3it plug-in file into the "System" folder on Windows, or the "FileMaker Extensions" folder on Macintosh, inside your FileMaker Pro 4.x or 5.x folder. If you have a previous version of POP3it, the installer will overwrite it.

Windows Users: The Installer will not run from inside an unzipping program like WinZip. You must unzip it to a folder before running the installer. Also, if you have WinZip and you have it set to automatically install applications that you download, you will not get the full benefit of this download. WinZip only installs the plug-in and does not show you the contents of the zip file, which contains the example databases and the documentation. You must unzip the POP3it zip file manually so that you can see the entire contents and not just the installer. If you set the WinZip "Wizard" interface to "Classic", it will not do this.

After you install the plug-in as described above, you can open FileMaker Pro and set the default preferences. To do this, go to Edit->Preferences->Application, click on the Plug-ins tab, and double-click the POP3it plug-in.

Basics

Once the Configuration Dialog is open, click the Basics tab (See Figure 1) where you can set the following values. POP3 Host is where you enter the domain name or IP address of your POP3 server. Username is where you can enter a default Username. (The Username is usually the part of your email address before the @ symbol.) Password is where you can enter a default Password.

POP3it 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 Username in your POP3it related script, POP3it will use the Default Username.

POP3it waits for certain amount of time for your POP3 host to respond. By default, this is set to 15 seconds. However, you can adjust the setting to fit your needs by entering a different value in the Connection Timeout.

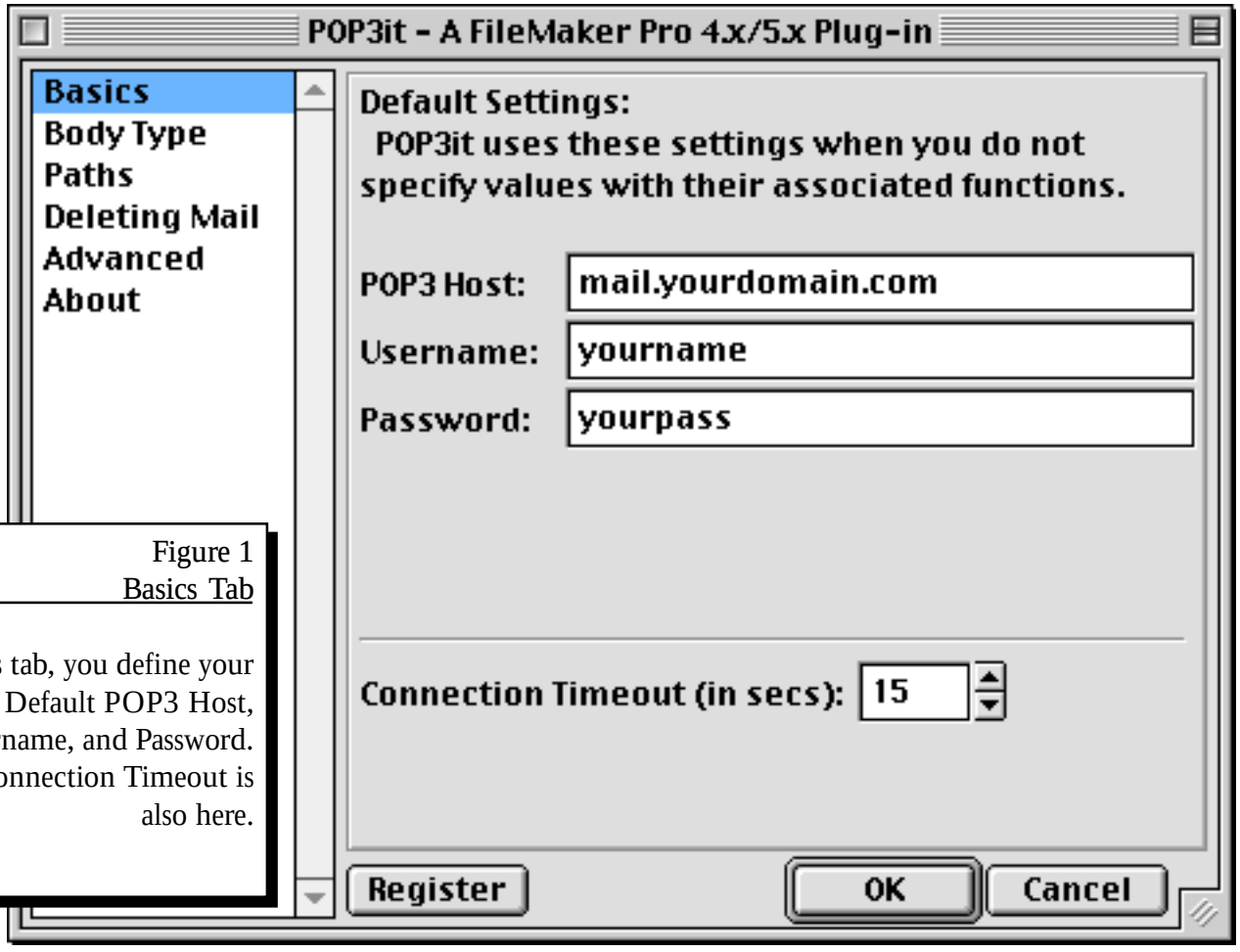

Figure 1
Basics Tab

On this tab, you define your Default POP3 Host, Username, and Password. The Connection Timeout is also here.

Body Type

Emails come in three main forms. There are Plain Text only emails, HTML only emails, and "multi-part/alternative" emails which have both a Plain Text part and an HTML part. You can specify which type of email you want POP3it to return when you call the POP3-CurrBody or POP3-CurrAll functions. When you do not specify which type you want when you call those functions, POP3it will use this default setting. If you select "Text", POP3it will only return a Plain Text body to you. If you select "HTML", POP3it will only return an HTML body to you. If you select "Both", POP3it will return a Plain Text body (if there is one) followed by an HTML body (if there is one). If you select "Text, HTML", POP3it will return a Plain Text body to you, unless there is not one, then it will return an HTML body to you. If you select "HTML, Text", POP3it will return an HTML body to you, unless there is not one, then it will return a Plain Text body to you.

It is possible for POP3it to return a blank body to you depending on the type of email and what type you ask for. For example, if you ask for "Text", and the email only has an HTML body, then POP3it will return a blank body to you. If you never want a blank body returned to you, check the Always return something checkbox. (Note: If there actually is nothing in the body of the email, Text or HTML, POP3it will still return a blank body because there was nothing to return.)

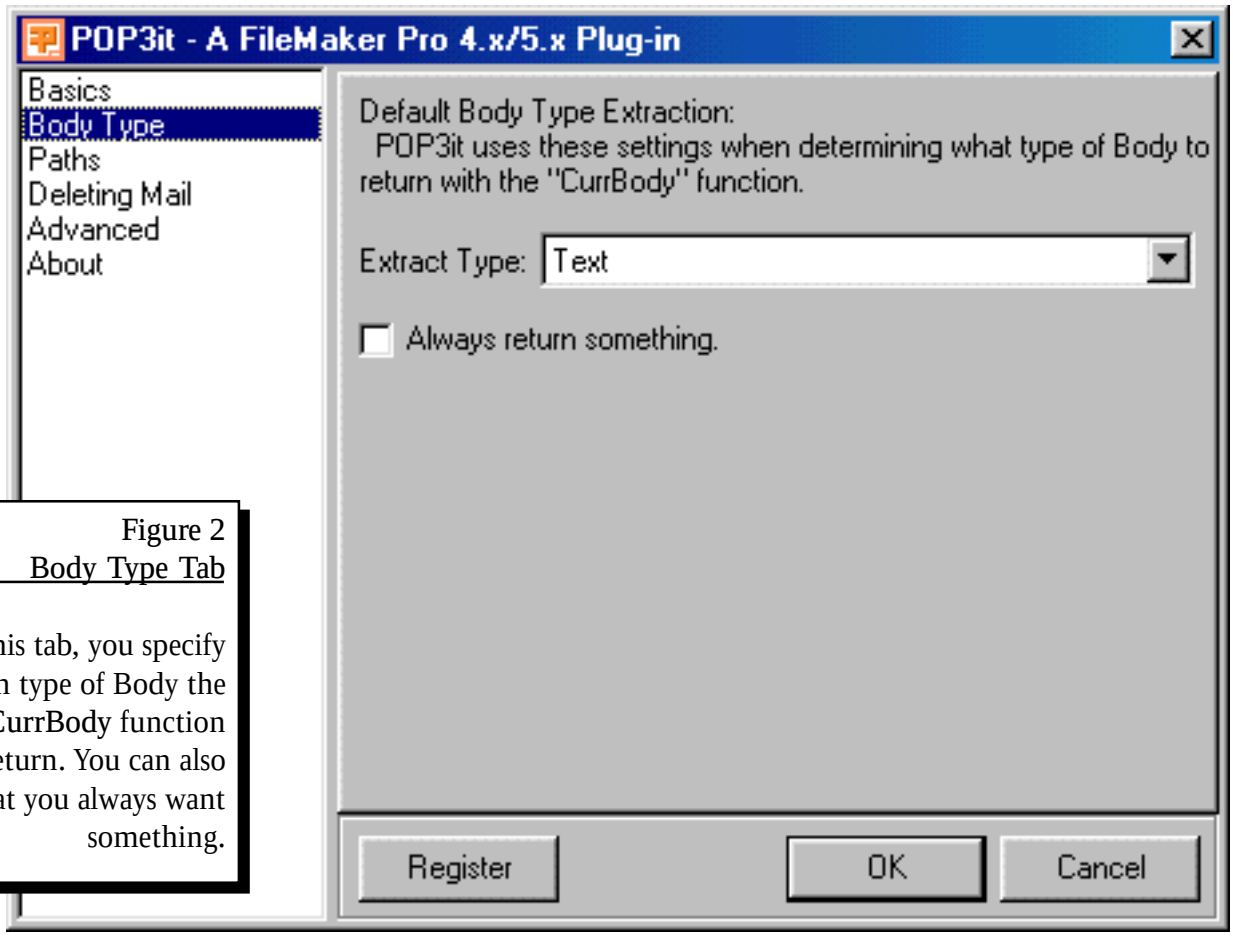

Figure 2
Body Type Tab

On this tab, you specify which type of Body the POP3-CurrBody function should return. You can also specify that you always want something.

Paths

If you do not specify where to put Attachments in your scripts, POP3it looks at these settings. Attachments is where you enter the default directory for saving attachments. HTML Export is where you enter the default directory for saving exported HTML emails. (You can use the ... button to browse for the directory.)

Deleting Mail

If you want POP3it to automatically delete email from the server you can select one of the following options. Choosing Never will tell POP3it to never automatically delete email. Choosing Before AcquireNext will tell POP3it to delete the email before it gets the next one. Choosing Before Disconnect will tell POP3it to delete the email before it closes the connection.

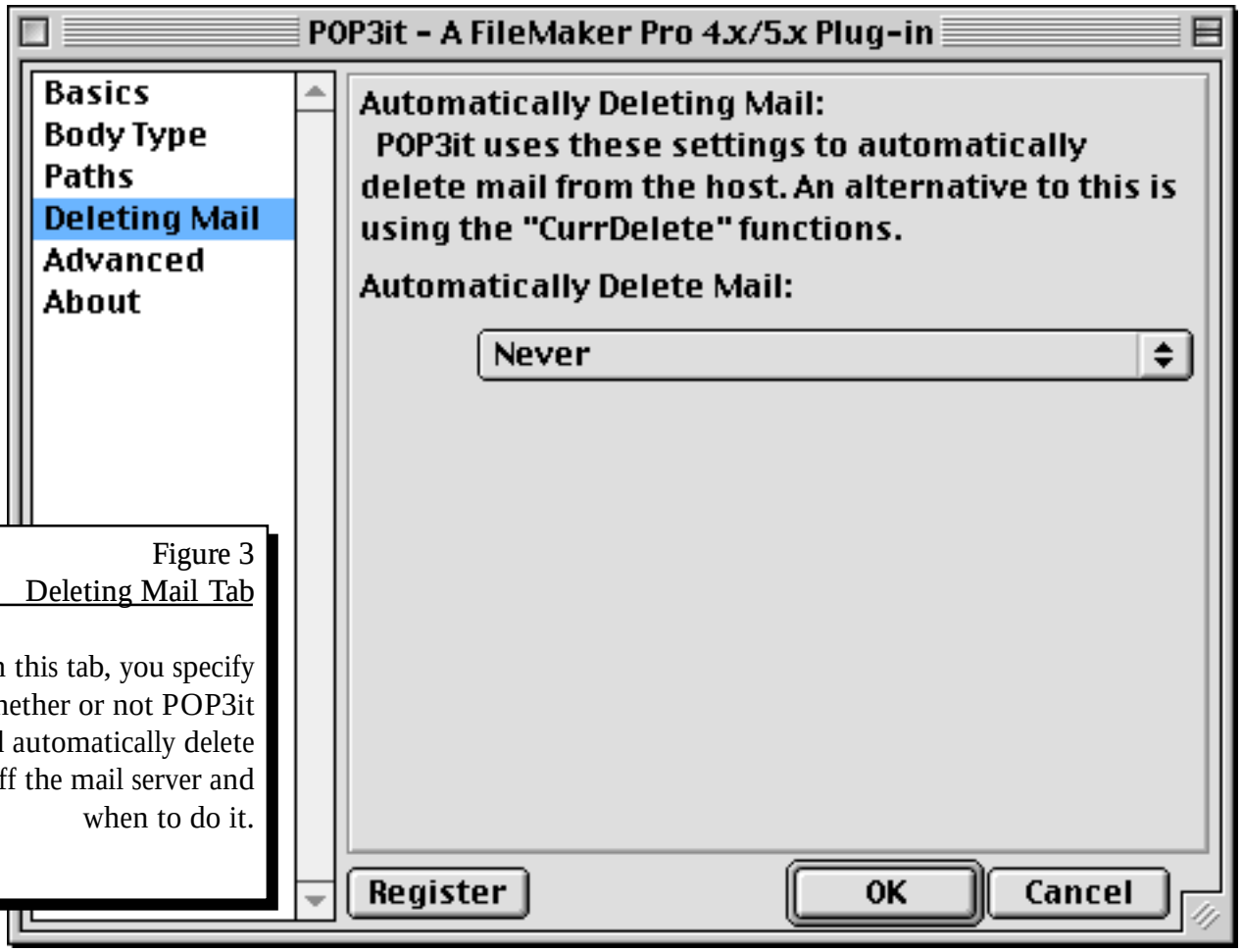

Figure 3
Deleting Mail Tab

On this tab, you specify whether or not POP3it should automatically delete email off the mail server and when to do it.

Advanced

New with POP3it 3.0 is the Advanced tab (See Figure 4). On this tab are advanced features that you probably will never need, and probably should not touch. However, if you know what you are doing, or if we instruct you to do so, you can change these settings.

The internal character set inside FileMaker Pro is not the more widely used character set used on the internet. Therefore POP3it must convert the FileMaker Pro character set before it can use it so that the characters appear correctly. For the most part, the only characters that are affected are the "High ASCII" characters which are usually accented characters and some mathematical symbols. If you do not want POP3it to do this character conversion, you can switch it off by checking the Disable Character Conversions checkbox (See Figure 8).

If you do want POP3it to do it's character conversions, but it appears to be doing the wrong character conversion for your localized system, you can change the Convert To setting to match your needs. For instance, if you have a Japanese version of FileMaker Pro and your Japanese characters are not appearing correctly, make sure the Convert To setting shows "ISO-2022-JP (Japanese)". When you first run POP3it, it should auto-detect which character set your computer is set to, and set the Convert To setting accordingly. However, if it does not auto-detect, you can change the setting.

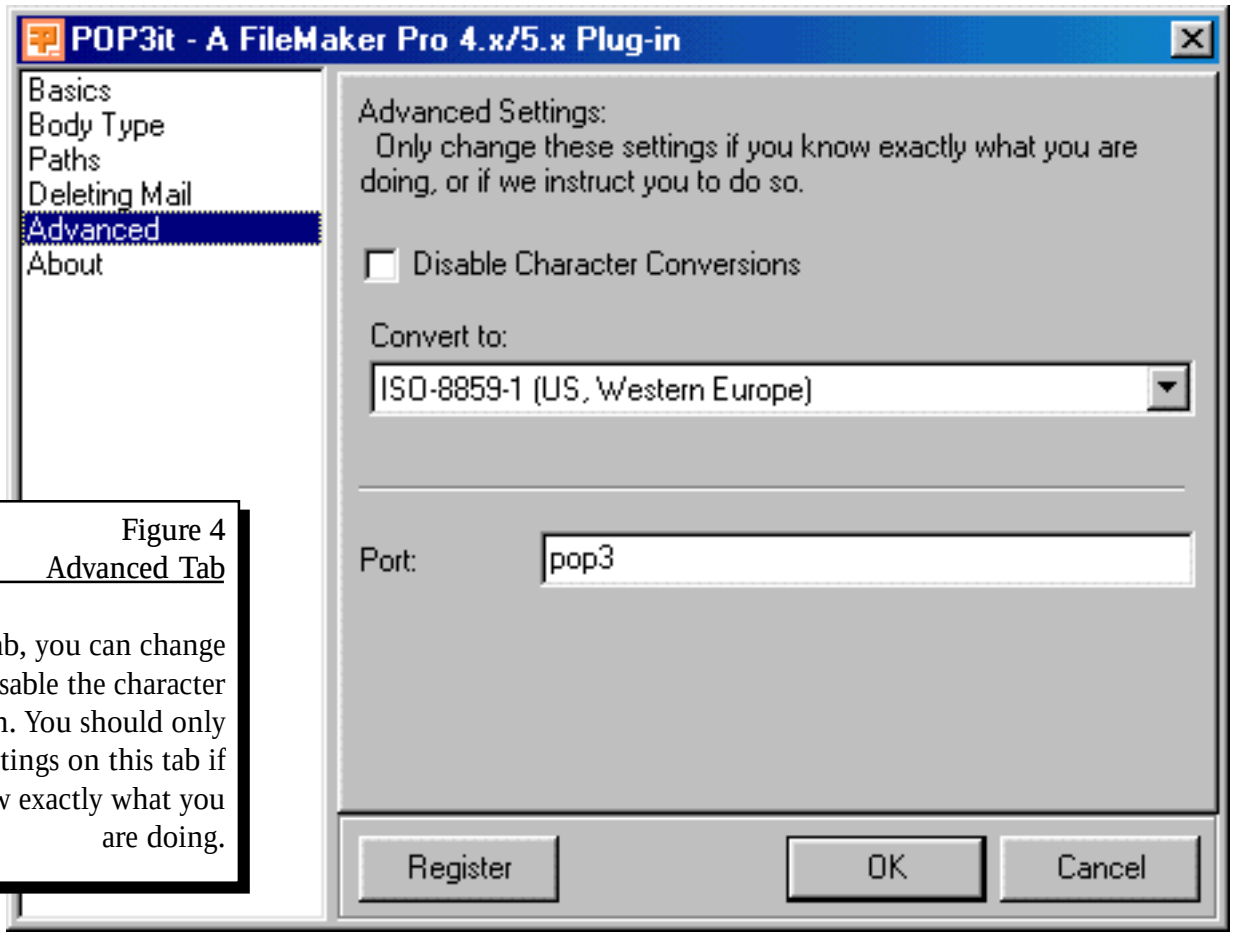

Figure 4
Advanced Tab

On this tab, you can change or disable the character conversion. You should only change settings on this tab if you know exactly what you are doing.

If your mail server requires you to connect on a different TCP/IP port than the standard POP3 port 110, you can change the Port setting to fit your needs.

About

The About tab simply reports which version of POP3it that you are using, and to whom this copy of POP3it is registered. You can also use the POP3-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 recent version of POP3it, please visit our website (http://www.pop3it.com/).

Registering POP3it

You can register your copy of POP3it from the Configuration Dialog once you have purchased it from our secure website. After you purchase POP3it, we will send you a registration number to register your copy by using the Register button in the Configuration Dialog (See Figure 5). We have also included the POP3-Register function so that developers can easily register POP3it with their bound solutions once they have purchased a Developer's License. For more information on purchasing POP3it and other exciting plug-ins, visit our website (http://www.cnsplug-ins.com/) and choose the Purchase link from the tool bar.

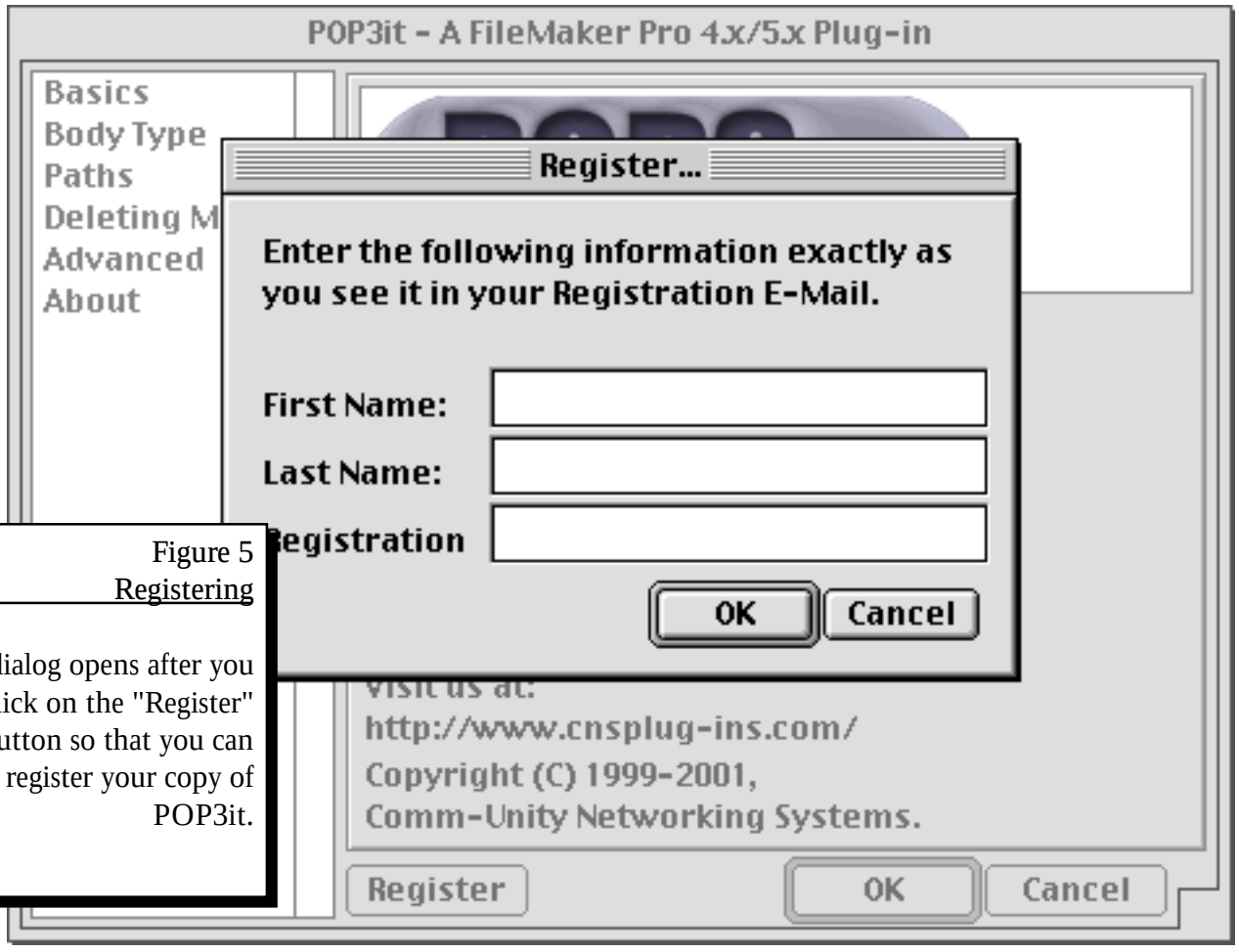

Figure 5
Registering

This dialog opens after you click on the "Register" button so that you can register your copy of POP3it.

How to Use FileMaker Pro Plug-ins

FileMaker, Inc. introduced a very simple plug-in architecture when they released FileMaker Pro 4.0. Originally intended to aid only in complex calculations, the plug-in architecture took off in ways that FileMaker, Inc. had not expected. Though there are now many different types of plug-ins, they all work the same basic way. By understanding the basics of plug-in interaction you will be able to understand how many developers approach plug-in development.

To start with, plug-ins are used by creating calculations. You can use calculations in many places within FileMaker Pro including a calculation field, the Set Field Script Step, the Paste Result Script Step, as well as a text field that has a calculated value. There are a couple of other ways to create calculations, but these are the major avenues that are currently being used for plug-ins. The most commonly used avenue for plug-ins is the Set Field Script Step. It is easy to use, plus, the plug-in can report its actions in the field that you are setting. If you have never used scripts before, you can find a complete explanation of Scripts and Script Steps in the FileMaker Pro Help Topics located in the Help menu.

Though the calculation dialog box is limited in space, FileMaker Pro has made it somewhat easier to deal with functions by using their groupings of functions in the top right corner of the calculation dialog. To view a list of all the External Functions of all the plug-ins currently installed, choose View by "External Functions". Once chosen, you should see the WebCompanion external functions (if WebCompanion is installed and enabled), and if you have POP3it installed (and enabled) you will see the POP3it External Functions listed as well. Choosing to View by "External Functions" can greatly increase the ease of your script writing because the External Functions that you need are right there at your finger tips. In fact, we highly recommend that you always choose External Functions by double clicking on them in this list. This will ensure that you have the correct spelling of the function.

FileMaker Pro's External Function looks like the following:

```
External("PluginNamePrefix-FunctionName", parameter)
```

For example take our POP3-Version function, which returns the version of POP3it that you are

using. When you double-click the External Function in the External Function list, it inserts this into your calculation:

External("POP3-Version", parameter)

To actually use the function though, you will have to give it a parameter. Since the POP3-Version function does not need any special information, you can just set it to the empty string ("") in order to use the function:

External("POP3-Version", "")

The parameter of the External Function is where you put the information for that function. For example, if you are setting your POP3 Host using the POP3-Host function, you could use a literal value like "mail.yourdomain.com":

External("POP3-Host", "mail.yourdomain.com")

You could also have a text field or a global field named "POP3Host" to hold your POP3 Host value. If you did, then you could use it like this:

External("POP3-Host", POP3Host)

The difference between using a field to hold the value and putting the literal value directly into the calculation is the double-quotation marks. This is because FileMaker Pro will interpret anything not in double-quotes as a field in your database.

You can also mix literal values with fields in your database by concatenating them together. You do this by inserting an ampersand (&) between the values. For example, if you have a field called "Auto Check Interval", you could use it in the POP3-AutoCheckEmail function like so:

External("POP3-AutoCheckEmail", Status(CurrentFileName) & "|Auto Check Mail|" &
Auto Check Interval)

FileMaker Pro will concatentate the values in that field with the value returned by Status(CurrentFileName) and the literal value "|Auto Check Mail|". If your "Auto Check Interval" field contained "15", and the database file name is "Email Downloader.fp3" then FileMaker Pro would turn the above into:

External("POP3-AutoCheckEmail", "Email Downloader.fp3" & "|Auto Check Mail|" &
"15")

And further into this:

External ("POP3-AutoCheckEmail", "Email Downloader.fp3|Auto Check Mail|15")

FileMaker Pro would then send all of that to POP3it which would use the parameter to set the Auto

Check Email feature.

Once you understand the above, you will be on your way to using plug-ins in no time. If you need more help understanding Scripts, Script Steps, and Calculations, consult the Help Files for FileMaker Pro located in the Help menu.

Example Databases Explained

A Typical Email Session

Let us think for a moment about the real world equivalent of a real Post Office Box compared to the Internet world of a POP3 email box. To receive your mail from the Post Office in the real world you must first go to the Post Office. Once there you locate your mail box and use a key to open the box. At this point you take your mail, close your mail box and leave the Post Office. Every once in a while you may receive a notice that you have a package to pick up that will not fit in your box, so you may have to pick up your package before you leave the Post Office.

In the Internet world, you can compare this process almost step by step. To receive your email on the Internet, you must first open a connection with your POP3 mail server and use your username and password to open your email box. At this point, you grab all of your email including any larger attachments, and then close the connection to your POP3 server.

There are several subtle differences between checking your email and getting mail in the real world, though. One difference is the fact that your email will be in your box until you delete it. You will want to make sure and read the following section called Issues to Think About When Using POP3it to understand some of the problems that you can run into when checking email.

Now that we have a basic understanding of how the process works, let us take it a step further and explain how to check an email account with POP3it and FileMaker Pro. Probably the best place to start is by walking through all of the steps it takes in order to get email from a POP3 mail server. For now, we will describe a simple path to receive email from your email server into FileMaker Pro, then we will give you an example script based on those steps.

Step 1: Set your POP3 Mail Host, Username, and Password.

Use the POP3-Host, POP3-Username, and POP3-Password functions to set your POP3 Host, Username, and Password. The POP3it External Function Reference at the end of this document fully describes these functions. When used properly, they will allow you to check multiple email accounts easily.

Step 2: Make a connection to your POP3 Mail Host.

To make a connection to your mail server you simply use the POP3-Connect function with the empty string ("") as the parameter.

Step 3: Set up the first email to download and see how many emails are available for download.

To set up the first email to download and see how many emails are available for download, we use the POP3-AcquireFirst function which returns how many emails are waiting and will also allow us to start working with the first email. If there are no emails on the mail server, this function will return a zero ("0"), which means we need to jump to Step 9 and close the POP3 connection. If there are one or more emails are available for download, then we need to continue on with Step 4 to get the emails.

Step 4: Enter A loop.

It is essential that we enter a loop here so that we will be able to get multiple emails. Since the POP3-AcquireFirst function sets up the first email to download, we are ready to get the information in the first email right when we enter the loop.

Step 5: Get current email information.

At this point, we can get the email in two different ways. The first way is to use the POP3-CurrAll function to get the entire email and put it into one field. This function will also download any email attachments to your hard drive in the directory that you have specified with the POP3-AttachPath or POP3-DlgAttachPath functions or in the Configuration Dialog. This way is the easiest, but it is not necessarily easy to work with if you need specific information in the email separated into multiple fields. The second way is to set fields using the other "Current" functions such as POP3-CurrUniqueID, POP3-CurrHeader, POP3-CurrDate, POP3-CurrTo, POP3-CurrFrom, POP3-CurrSubject, POP3-CurrBody, and POP3-CurrAttach. This is somewhat easier because POP3it has extracted the major email fields already for you. All you have to do is set the appropriate fields using the above functions. This is also the best place to run any sub-scripts you may have based on the content of the email. For example, you could check the subject of the email for a keyword, and run a script based on your findings.

Step 6: Delete the current email.

Technically, you do not have to delete the current email at this time. However, it is much easier to always delete the emails so you do not have to check for duplicate emails each time you check your email account. When you are first testing your POP3it script, you may choose to skip deleting the emails until you have the script working the way you want it to, otherwise, you may have to keep sending yourself test emails.

Step 7: Check to see if there are more emails to download. / Exit loop.

The next step is to see if there are any more emails to download. To do this, use the POP3-AcquireNext function. Since POP3-AcquireFirst has already given us our first email, POP3-AcquireNext will give us the second email. Throughout the loop this function will continue to give us the next email, incremented by one, until there are no emails left to download. When there are no emails left to download, this function will give us a value of zero ("0"), thus giving a test to break out of the loop. If POP3-AcquireNext does return a zero, we will need to break out of the loop and jump to Step 9 to close the POP3 connection.

Step 8: Retrieve the next email.

At this point we are at the end of the loop. If we have gotten this far, we have already retrieved at least one email, and there is at least one more email to download since we made it past Step 7. So, the loop will start over with Step 5 and continue to loop until Step 7 proves that we have no more emails.

Step 9: Close the connection.

This last step is simple, but crucial. If you do not close the POP3 connection with your host you can cause problems with both POP3it and with your POP3 mail host. This is one of the pitfalls of using a POP3 host. By its nature, you have to manually open the connection and manually close it. For more information on how things can go awry and things to look out for, read the next section called Issues to Think About When Using POP3it. With that said, to close the POP3 connection simply use the POP3-Disconnect function with the empty string ("") as the parameter.

Issues to Think About When Using POP3it

There are several issues to think about when using POP3it to check an email account. The first major issue is to make sure that every time you open a connection with your POP3 server, that you also close that connection. This may seem quite obvious, however, if you decide to use the Paste Result Script Step to close your connection, and the field that you are pasting the result to is not on the current layout, FileMaker Pro will not activate the function and POP3it will not close the connection. For this reason, it is wise to either make sure that the field you are working with is on the current layout, or that you use the Set Field Script Step, because the Set Field Script Step does not require the result field be on the current layout.

You can run into another problem if you decide not to delete each message from the server when you check email. If you do not build in some kind of email UniqueID checking, you will receive the same email messages over and over again. An example of checking the UniqueID of an email is provided in the POP3it Example database and the POP3it Example Explained section later in this documentation.

Another issue you may want to think about is error checking. POP3it has several error codes that you can use. Refer to the section Understanding Error Responses for help with POP3it error codes.

Quick Start Example Explained

The Quick Start Example database is a fairly simple database that should help you understand the basics of receiving email into FileMaker Pro with POP3it. This database contains the following seven fields:

Field Name	Field Type
POP3 Host	Global (Text)
Username	Global (Text)
Password	Global (Text)
Attach Path	Global (Text)
Full Email	Text
Setup Result	Global (Text)
Curr Result	Global (Text)

The first three fields should be self explanatory. You can fill them in appropriately. The fourth field, Attach Path, will hold the location on the hard drive where you want POP3it to download the attachments. For Macintosh users, fill this field in with an appropriate path like "Macintosh HD:POP3it Attachments:". For Windows users, fill this field in with an appropriate path like "C:\POP3it Attachments\". The Full Email field will contain the entire email including all headers, email content, and attachment path and file names. The final two fields are used in downloading the email from your mail server as explained below.

Let us look at the scripts in this database. Go to the Script menu and select ScriptMaker... Select the Check Mail script, and press the Edit button. The Script Definition for "Check Mail" dialog should open. In the big text area on the right you should see the content of the script, which should look like the following:

```
 1  Enter Browse Mode []
 2  Go to Layout ["Example"]
 3  Comment ["First we need to setup our Host, Username, and Passwo..."]
 4  Set Field ["Setup Result", "External("POP3-Host", POP3 Host) & ..."]
 5  If ["PatternCount(Setup Result, "ERROR") > 0"]
 6    Show Message ["There was an error setting up POP3it or connec..."]
 7    Exit Script
 8  End If
 9  Comment ["Connection was successful, so let's see if there is a..."]
10  Set Field ["Curr Result", "External("POP3-AcquireFirst", "")" ]
11  If ["LeftWords(Curr Result, 1) = "ERROR""]
12    Set Field ["Setup Result", "Setup Result & "¶" & External("PO..."]
13    Show Message ["There was an error getting the first email. Pl..."]
14    Exit Script
15  End If
16  If ["Trim(Curr Result) = "0""]
17    Set Field ["Setup Result", "Setup Result & "¶" & External("PO..."]
18    Show Message ["There is no email on the server to download."]
19    Exit Script
20  End If
21  Comment ["There is at least one email on the server, so now we ..."]
22  Loop
23    Comment ["Make a new record for the new email, and download t..."]
24    New Record/Request
25    Set Field ["Full Email", "External("POP3-CurrAll", "Text,HTML")"]
26    Comment ["Now we have the current email, so go get the next."]
27    Set Field ["Curr Result", "External("POP3-AcquireNext", "")"]
28    Exit Loop If ["(Trim(Curr Result) = "0") or (LeftWords(Curr R..."]
29  End Loop
30  Comment ["We have now downloaded all of the email, so disconnec..."]
31  Set Field ["Setup Result", "Setup Result & "¶" & External("POP3..."]
32  Show Message ["You have new mail!"]
```

Let us figure out what each of these Script Steps are doing. Lines 1 and 2 simply make sure you are in the correct mode and on the correct layout. Line 3 is a Comment Script Step to remind ourselves what we are doing, which is that we are about to set up POP3it and try to connect to the mail server.

Line 4 is the first interesting line. This line contains a Set Field Script Step that will be setting our Setup Result field. Select this Set Field Script Step and look near the bottom of the dialog, where some buttons should have appeared. Press the Specify... button. Now the Specify Calculation dialog is open. You should see several External() functions in the large text area. These should look familiar if you read the How To Use FileMaker Pro Plug-ins section earlier in this documentation. You should see the following:

```
A  External("POP3-Host", POP3 Host) & "¶" &
B  External("POP3-Username", Username) & "¶" &
C  External("POP3-Password", Password) & "¶" &
D  External("POP3-AttachPath", Attach Path) & "¶" &
E  External("POP3-Connect", "")
```

Line A tells POP3it the POP3 Host domain name that we will be connecting to. Line B tells POP3it the Username for the account that we will be logging in to, and Line C tells POP3it the Password for that account. Line D tells POP3it where to save attachments from the emails we will be downloading. Finally, Line E tells POP3it to try to connect to the mail server.

Looking back at the Check Mail script and Line 5 you see an If Script Step. This line looks for the word "ERROR" in the Setup Result field that we just set with Line 4. If it finds an error, then Line 6 informs the user that there was an error, and Line 7 uses an Exit Script Script Step to abort the script. Since there was an error setting up or connecting to the mail server, then there is no way for us to download any email, which is why we have to exit the script.

Line 9 uses another Comment Script Step to say that at this point we are successfully connected and that we will now try to get the first email and the total email count. Line 10 does exactly this by using a Set Field Script Step and calling the POP3-AcquireFirst function.

Line 11 looks at the Curr Result field that was just set by Line 10 with the results of the POP3-AcquireFirst function. If Line 11 finds the word ERROR at the beginning, then it knows that POP3-AcquireFirst had an error. If POP3-AcquireFirst had an error, then there is something wrong, so there is no way to continue downloading email and we need to abort. Line 12 uses the POP3-Disconnect function to disconnect from the mail server, while Line 13 will tell the user that there was an error. Line 14 will abort the script with the Exit Script Script Step.

If there was no error with the POP3-AcquireFirst function, then we need to see if there is any email on the mail server to download. Line 16 will see if the Curr Result field, that was set by th Set Field Script Step in Line 10, contains "0". If it does, then there is no email on the server to download, so Line 17 will disconnect from the mail server. Line 18 will tell the user that there is no email on the server, and Line 19 will abort the script.

If we get to Line 21, then there is at least one email on the server to download. Line 22 starts our email downloading loop to download all available email on the mail server. Line 23 uses another Comment Script Step to remind us what we are doing.

Line 24 uses the New Record/Request Script Step to create a new record for the new email we are about to download. If we did not create a new record, then we would be constantly overwriting the same record and it would look like we only downloaded one email. Line 25 uses a Set Field Script Step to set our Full Email field with the entire email downloaded with the POP3-CurrAll function.

Line 26 reminds us that we have done all we need to with the current email, so we must get the next. This is done in Line 27 which uses the POP3-AcquireNext function. Line 28 will inspect the Curr Result field that we just set in Line 27. If we find a "0" in the field, then we know we have downloaded all of the email available on the server and must exit the email downloading loop. Line 28 also makes sure that the POP3-AcquireNext function did not have an error, and if it did, will also exit the email downloading loop.

Line 30 uses a Comment Script Step to remind us that at this point we have downloaded all the email and exited the email downloading loop. We now need to disconnect from the mail server using the POP3-Disconnect function in Line 31, and inform the user that they have new email in Line 32.

That concludes the Check Mail script. As you can see, not much is required to download email into your database using POP3it. The majority of the lines above are error checking, and once you look past all of those, you see that downloading email into your database with POP3it is a fairly simple process requiring only a few fields and a few Script Steps. Once you understand this database and its Check Mail script, you can take a look at the more advanced POP3it Example Database explained next in the documentation.

POP3it Example Explained

This section of the documentation will explain the Check Mail script and its related scripts in the POP3it Example database. This set of scripts is a little more advanced than the Check Mail 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 POP3it.

There are twenty-two fields in the POP3it Example database that are used by the Check Mail scripts as explained in this documentation. These fields are the following:

Field Name	Field Type
Host	Global Text
Username	Global Text
Password	Global Text
APOP	Global Text
Attach Path	Global Text
Delete Mail	Global Text
Delete Mail Day Count	Global Number
Setup Result	Global Text
Curr Result	Global Text
Delete Mail Date	Global Date
From	Text
To	Text
CC	Text
Date Text	Text
Subject	Text
Full Headers	Text
Email	Text
Attachments	Text
Date	Calculation
UniqueID	Text
Curr UniqueID	Global Text
Record Count	Global Number

The first three fields are the exact same as the Quick Start Example database, so those should be familiar. The fourth field, APOP, will hold an "On" or "Off" value depending on if the user's mail server requires APOP authentication. The Attach Path field is the same as in the Quick Start Example database. The sixth field, Delete Mail, holds a "Yes" or "No" value depending on if the user wants to delete all email off the mail server after downloading it. The Delete Mail Day Count field will hold the number of days the user wishes to leave the email on the server before deleting it. Setup Result and Curr Result are used when downloading the email just like in the Quick Start Example database. The Delete Mail Date field is used when testing to see if an email on the mail server is older than the number of days defined in the Delete Mail Day Count field.

The next few fields will contain the actual content of the emails that will be downloaded. The From field will contain the From email address from the email. The To and CC fields will contain the To and CC email addresses from the email. The Date Text field will contain the date in text format from the email. The Subject field will contain the subject of the email, and the Full Headers field will contain the entire Header section of the email. The Email field will hold the actual body or content of each email, while the Attachments field will hold the paths and filenames of the attachments that POP3it decoded and saved from the emails.

The Date field is a calculated field that is based on the Date Text field. When you use the POP3-CurrDate function to return the text date from the email, POP3it forces that text date to be compliant with the RFC822 internet standard. This means that the text date will always be of the same fixed form allowing you to have a simple calculation to turn it into a FileMaker Pro Date or Time field. This Date field uses the following calculation to turn the Date Text field into an actual FileMaker Pro Date:

```
A  TextToDate(
B    Case(
C      Upper(Middle( Date Text, 9, 3)) = "JAN", "01",
D      Upper(Middle( Date Text, 9, 3)) = "FEB", "02",
E      Upper(Middle( Date Text, 9, 3)) = "MAR", "03",
F      Upper(Middle( Date Text, 9, 3)) = "APR", "04",
G      Upper(Middle( Date Text, 9, 3)) = "MAY", "05",
H      Upper(Middle( Date Text, 9, 3)) = "JUN", "06",
I      Upper(Middle( Date Text, 9, 3)) = "JUL", "07",
J      Upper(Middle( Date Text, 9, 3)) = "AUG", "08",
K      Upper(Middle( Date Text, 9, 3)) = "SEP", "09",
L      Upper(Middle( Date Text, 9, 3)) = "OCT", "10",
M      Upper(Middle( Date Text, 9, 3)) = "NOV", "11",
N      Upper(Middle( Date Text, 9, 3)) = "DEC", "12",
O      "01"
P    ) &
Q    "/" &
R    Middle( Date Text, 6, 2) &
S    "/" &
T    Middle( Date Text, 13, 4)
U  )
```

A standard RFC822 date will look like "Mon, 06 Aug 2001 03:50:33 -0500". Lines B through P use a Case to look at the 9th, 10th, and 11th position to determine what month it is. In our example, it would be looking at "Aug", which stands for August. Line R will look at the 6th and 7th position for the day of the month. In our example, it would be looking at "06". Line T looks at the 13th, 14th, 15th, and 16th position, which should be the year. In our example, it would be looking at "2001". Finally, Line A uses the TextToDate function and the calculated "08/06/2001" value to construct a FileMaker Pro Date.

Now back to our database fields and the UniqueID field. This field will contain the Unique Identifier of each of the emails. This UniqueID is useful for determining if we already have a specific email in our database to make sure we do not download an email a second time. The Curr UniqueID field is used in a self-relationship for this UniqueID test that keeps us from downloading duplicate emails.

Finally, the Record Count field is used to determine if we have downloaded any new email in our scripts.

Now, let us look at the Check Mail script:

```
 1  Enter Browse Mode []
 2  Go to Layout ["List View"]
 3  Comment ["Save our current record count so that we can determin..."]
 4  Set Field ["Record Count", "Status(CurrentRecordCount)"]
 5  Comment ["Go set everything up and try to connect."]
 6  Perform Script [Sub-scripts, " \_ Setup and Connect"]
 7  If ["PatternCount(Setup Result, "ERROR") > 0"]
 8    Show Message ["There was an error setting up POP3it or connec..."]
 9    Exit Script
10  End If
11  Comment ["Connection was successful, so let's go and try to get..."]
12  Perform Script [Sub-scripts, " \_ Start First Message"]
13  If ["LeftWords(Curr Result, 1) = "ERROR""]
14    Perform Script [Sub-scripts, " \_ Disconnect"]
15    Show Message ["There was an error getting the first message. ..."]
16    Exit Script
17  End If
18  If ["Trim(Curr Result) = 0"]
19    Perform Script [Sub-Scripts, " \_ Disconnect"]
20    Show Message ["There is no new mail on the server."]
21    Exit Script
22  End If
23  Comment ["There is at least one email on the server, so go run ..."]
24  Perform Script [Sub-scripts, " \_ Get Messages Loop"]
25  Comment ["We are done downloading all the email, so disconnect."]
26  Perform Script [Sub-scripts, " \_ Disconnect"]
27  Comment ["Now check our new current record count with our previ..."]
28  If ["Record Count < Status(CurrentRecordCount)"]
29    Show Message ["You have new mail!"]
30  Else
31    Show Message ["There is no new mail on the server."]
32  End If
```

Lines 1 and 2 simply make sure you are in the correct mode and on the correct layout. Line 3 uses a Comment Script Step to remind us that we are doing next. Line 4 sets our Record Count field with the current record count so we know how many records we are starting with.

Line 6 calls our Setup and Connect script to set the POP3it settings and try to connect. This script contains a single Set Field Script Step with this calculation:

```
A  External("POP3-Host", Host) & "¶" &
B  External("POP3-Username", Username) & "¶" &
C  External("POP3-Password", Password) & "¶" &
D  External("POP3-APOP", APOP) & "¶" &
E  External("POP3-AttachPath", Attach Path) & "¶" &
F  External("POP3-Connect", "")
```

Line A sets the POP3 Host domain name to connect to. Lines B and C set the Username and Password for the account that we will be downloading email from. Line D will tell POP3it whether or not the users account requires the APOP authentication method. Line E will set the location on the hard drive for the attachments, and finally Line F will try to connect to the mail server. (Note: This is not the exact calculation in the POP3it Example database, but instead has been slightly simplified for the purpose of this documentation.)

Now back to our Check Mail script and Line 7. This line looks in the Setup Result field that was set in the Setup and Connect script in Line 6. If it finds the word "ERROR", then it knows that the Setup and Connect script had an error setting up POP3it or connecting to the mail server. If there was an error, then Line 8 will inform the user that there was an error and Line 9 will abort the script with an Exit Script Script Step.

If we get to Line 11, then we have successfully connected to the mail server. Line 12 calls our Start First Message script to set up the first email for downloading and to return the number of emails available on the mail server. This script has a single Set Field Script Step with this calculation:

```
External("POP3-AcquireFirst", "")
```

This simply uses the POP3-AcquireFirst to set up the first email for download and to return the number of emails available.

Line 13 of our Check Mail script will look at the Curr Result field which was just set by the Start First Message script on Line 12. If Line 13 finds that the Curr Result field contains an error, then Line 14 will disconnect us from the mail server by calling the Disconnect script. This script contains a single Set Field Script Step with the following calculation:

```
A  Setup Result & "¶" &
B  External("POP3-Disconnect", "")
```

Line A will keep the previous value of the Setup Result field and add on the result from the POP3-Disconnect function in Line B.

Now back to our Check Mail Script, you see that Line 15 will inform the user that there was an error getting the first email, and Line 16 will abort the script.

Line 18 also looks at the Curr Result field, but instead of looking for an error, it looks to see if the field contains "0". If the field does contain "0", then there is no email on the server to download. Line 19 would then call our Disconnect script to disconnect from the mail server, while Line 20 will inform the user that there was no email on the server to download. Line 21 would then abort the script with an Exit Script Script Step.

If we get to Line 23, then we have successfully connected to the mail server and there is at least one email on the mail server to download. So, Line 24 will call our Get Messages Loop script which will download all of our email. This script is explained a little later.

Line 25 uses another Comment Script Step to remind us that at this point we have downloaded all the email from the mail server and we need to disconnect from the mail server, which is what Line 26 does. Line 28 will then see if our stored record count from Line 4 is less than the current record count. If so, then Line 29 tells our user that they have new email. Otherwise, Line 31 tells the user that there was no new email on the mail server.

That concludes the Check Mail script for the POP3it Example database. If you compare it to the Check Mail script in the Quick Start Example database, you will see that there really is not that much difference.

Now, let us take a look at that Get Messages Loop script, which is very different than the email downloading loop found in the Check Mail script in the Quick Start Example database:

```
 1  Loop
 2    Comment ["First we need to grab the UniqueID of this email an..."]
 3    Set Field ["Curr UniqueID", "External("POP3-CurrUniqueID", "")"]
 4    If ["Count(Self UniqueID Test::UniqueID) = 0"]
 5      Comment ["If the self-relationship contains no records, the..."]
 6      New Record/Request
 7      Perform Script [Sub-scripts, "    \_ Get Current Message"]
 8      Comment ["If the user wants to delete all email after downl..."]
 9      If ["Delete Mail = "YES""]
10        Set Field ["Curr Result", "External("POP3-CurrDelete", "")"]
11      End If
12    Else
13      Comment ["Our self-relationship found a matching record, so..."]
14      If ["(Delete Mail = "NO") and (Delete Mail Day Count > 0)"]
15        Comment ["The user wants to delete all email after a few ..."]
16        Set Field ["Curr Result", "External("POP3-CurrDate", "")"]
17        Set Field ["Delete Mail Date", "TextToDate( Case( Upper(M..."]
18        If ["Delete Mail Date < (Status(CurrentDate) - Delete Mai..."]
19          Set Field ["Curr Result", "External("POP3-CurrDelete", "")"]
20        End If
21      End If
22    End If
23    Comment ["We've done all we need to with the Current Email, s..."]
24    Set Field ["Curr Result", "External("POP3-AcquireNext", "")"]
25    Exit Loop If ["(Trim(Curr Result) = "0") or (LeftWords(Curr R..."]
26  End Loop
```

Line 1 begins our loop, while Line 2 reminds us what we are about to do. Line 3 uses the POP3-CurrUniqueID function to get the Unique Identifier of the current email. It downloads this UniqueID into the Curr UniqueID field. This field is used in a self-relationship to determine if the email has already been downloaded or not. The self-relationship pairs this Curr UniqueID field against the UniqueID field which is different for each record. If the current email has already been downloaded, then the UniqueID of one of the records will contain the UniqueID of the email, and the self-relationship will match the UniqueID in the Curr UniqueID field with the one in the record, making the self-relationship valid.

If the self-relationship is not valid, then the Count function in Line 4 will return zero, meaning the email has not been downloaded. So, Line 6 creates a new record for the new email, and Line 7 calls the Get Current Message script to download the email. We will look at the Get Current Message script a little later.

Line 9 looks at the Delete Mail field to see if the user wants to delete all the email after downloading it from the mail server. If so, the Line 10 will use the POP3-CurrDelete function to delete the email.

If the self-relationship from Line 3 is valid, then the Count function in Line 4 will return something other than zero, which means we already have this email in the database. So, Line 14 will look at the Delete Mail and Delete Mail Day Count fields to see if the user wants to delete the email after the email is a certain number of days old. If so, then Line 16 will get the current date of the email. Line 17 uses the same calculation, based on a different field, as the Date Calculated field we looked at

above, to turn the email date we just downloaded into a FileMaker Pro Date we can use. Line 18 checks to see if the email date is less than the current date minus the user defined number of days in the Delete Mail Day Count field.

Line 23 reminds us that we have done everything we need to with the current email, so we need to get the next email. Line 24 uses the POP3-AcquireNext function to get the next email and Line 25 checks the result of the POP3-AcquireNext function to see if we have downloaded all the email or if there was an error.

Let us take a look at the final script, which is the Get Current Message script:

```
 1  Comment ["Get all the emails fields."]
 2  Set Field ["UniqueID", "External("POP3-CurrUniqueID", "")"]
 3  Set Field ["From", "External("POP3-CurrFrom", "")"]
 4  Set Field ["To", "External("POP3-CurrTo", "")]
 5  Set Field ["Date Text", "External("POP3-CurrDate", "")"]
 6  Set Field ["Subject", "External("POP3-CurrSubject", "")"]
 7  Set Field ["Full Headers", "External("POP3-CurrHeader", "")"]
 8  Set Field ["CC", "External("POP3-CurrHeader", "CC")"]
 9  Set Field ["Email", "External("POP3-CurrBody", "")"]
10  Set Field ["Attachments", "External("POP3-CurrAttach", "")"]
```

This script should be fairly self-explanatory. It fills in each of the fields with their appropriate values from the current email. Lines 7 and 8 may be a little confusing, so we will look at them.

Line 7 uses the POP3-CurrHeader function to return the entire Header section of the email. It does this by specifying the empty string ("") as the parameter. If you do not specify a parameter, then the POP3-CurrHeader function will return the entire Header section of the email.

Line 8 also uses the POP3-CurrHeader function, but this time specifies the parameter "CC". Since there is not a POP3-CurrCC function, then we need a way to return the CC email addresses if we want them. If you give the POP3-CurrHeader function a specific header to search for and extract, like "CC", then it will only return the portion of the Header section that you want. In fact, you can use the POP3-CurrHeader function to return any header from the Header section of the email, even headers for which there are already functions for, like POP3-CurrTo or POP3-CurrSubject.

Note: This script is not exactly like the actual script in the POP3it Example database. It has been simplified for this documentation.

This concludes the explanation of the POP3it Example database. By now you hopefully have a full understanding of how to download emails into your FileMaker Pro database using POP3it. For help with all of the functions available in POP3it, please see the POP3it External Function Reference later in this documentation. Once you have a complete understanding of how the POP3it Example database works, see if you can find the potential problems in the scripts. Need a hint? What happens to the UniqueID check and the "delete after so many days" check if you deleted an email in your database?

Please keep in mind that the example databases we provide are just that, examples. They are by no means the only way to use POP3it. We encourage you to play with the example databases, check out some of the other functions available in POP3it, 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 POP3it at http://www.pop3it.com/. If you still cannot find answers to your questions, please feel free to email us at pop3it@comm-unity.net.

External Function Reference and Contact Information

POP3it External Function Reference

The following is a list of all available functions and a complete description of each. Note that there are a few functions that can have default values set in the Configuration Dialog. See the Installation and Configuration section for a complete understanding of how the Configuration Dialog works.

For general information on how the External() functions work and how you can use them in your calculations, please see the How to use FileMaker Pro Plug-ins section earlier in this document.

Functions with an asterisk (*) are new with version 3.0. Functions with a pound sign (#) have changed with version 3.0

POP3-Version

This function returns the current version of POP3it 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.

Examples:

```
External("POP3-Version", "")
External("POP3-Version", "CONFIGURE")
External("POP3-Version", "ABOUT")
```

POP3-Register

This function allows you to register your copy of POP3it 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. This function looks like the following:

External("POP3-Register", "First Name|Last name|Serial Number").

The parameter consists of your first name, last name, and serial number all separated by the pipe character (|). ("|" is created by typing shift-backslash.)

POP3-ShowStatus *

Use this function to show a status window that displays information about what the plug-in is currently doing. When you are receiving an email, the status window will show you what part of the email it is currently receiving 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 POP3-HideStatus in a "close" script that you define in Edit->Preferences->Document of your main database.

Examples:

```
External("POP3-ShowStatus", "")
External("POP3-ShowStatus", "700,100")
External("POP3-ShowStatus", "-1,600")
```

See Also:

POP3-HideStatus (34)
POP3-MoveStatus (35)

POP3-HideStatus *

Use this function to hide the status window that you had previously shown with the POP3-ShowStatus function. You must call this function to hide the status window before closing the FileMaker Pro application (see POP3-ShowStatus above). 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("POP3-HideStatus", "")
```

See Also:

POP3-ShowStatus (33)
POP3-MoveStatus (35)

POP3-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 POP3-ShowStatus function above. 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.

Examples:

External("POP3-MoveStatus", "700,100")
External("POP3-MoveStatus", "-1,600")
External("POP3-MoveStatus", "-1,-1")

See Also:

POP3-ShowStatus (33)
POP3-HideStatus (35)

POP3-Host

The host is the domain name or IP address of the POP3 mail server you are connecting to. You can assign the host in your scripts, or set up a Default Host in the Configuration Dialog. POP3-Connect will return an error if no host has been set. If you assign a host with this function, POP3it will ignore the Default Host in the Configuration Dialog.

Example:

External("POP3-Host", "mail.yourdomain.com")

See Also:

POP3-Port (35)
POP3-Username (36)
POP3-Password (36)
POP3-Connect (38)
POP3-Disconnect (39)

POP3-Port *

Use this function to specify an alternate TCP/IP port for POP3it to connect to on your mail server. POP3it defaults to the standard POP3 TCP/IP port: 110. You will most likely never need to use this function.

Example:

External("POP3-Port", "16384")

See Also:

POP3-Host (35)

POP3-Username

Use the POP3-Username function to assign the username for your email account on the mail server. Your username is usually the first portion of your email address, for example, if your email address were "bob47@yahoo.com", your username would be "bob47".

Example:

External("POP3-Username", "frank")

See Also:

POP3-Host (35)
POP3-Password (36)
POP3-APOP (36)
POP3-Connect (38)

POP3-Password

Use this function to assign the password for your email account on the mail server.

Example:

External("POP3-Password", "love")

See Also:

POP3-Host (35)
POP3-Username (36)
POP3-APOP (36)
POP3-Connect (38)

POP-APOP *

If your mail server requires that you use the APOP authentication method for logging into your mail server, than you can use this function to turn it on or off.

Examples:

External("POP3-APOP", "ON")
External("POP3-APOP", "OFF")

See Also:

POP3-Host (35)
POP3-Username (36)
POP3-Password (36)
POP3-Connect (38)

POP3-PathToDB *

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 use this path to save your Attachments and HTML Exports. You must provide the name of the database as the parameter.

Example:

External("POP3-PathToDB", Status(CurrentFileName))
External("POP3-PathToDB", "my_database.fp3")

See Also:

POP3-AttachPath (37)
POP3-HTMLExportPath (38)

POP3-AttachPath

This function allows you to dynamically set the path on your hard drive where you want to save incoming attachments. You may want to use this function if you are checking multiple email accounts and want an attachments folder for each account.

Examples:

External("POP3-AttachPath", "C:\Attachments\")
External("POP3-AttachPath", "Macintosh HD:Attachments:")

See Also:

POP3-DlgAttachPath (37)
POP3-HTMLExportPath (38)
POP3-DlgHTMLExportPath (38)

POP3-DlgAttachPath

The POP3-DlgAttachPath function does the same thing as POP3-AttachPath, except that it presents you with a dialog so that you can select the path for the attachments. The parameter should be the empty string ("").

Example:

External("POP3-DlgAttachPath", "")

See Also:

POP3-AttachPath (37)
POP3-HTMLExportPath (38)
POP3-DlgHTMLExportPath (38)

POP3-HTMLExportPath

This function allows you to dynamically set the path on your hard drive where you want to save HTML exports.

Examples:

External("POP3-AttachPath", "C:\HTMLExports\")
External("POP3-AttachPath", "Macintosh HD:HTMLExports:")

See Also:

POP3-AttachPath (37)
POP3-DlgAttachPath (37)
POP3-DlgHTMLExportPath (38)

POP3-DlgHTMLExportPath

The POP3-DlgHTMLExportPath function does the same thing as POP3-HTMLExportPath, except that it presents you with a dialog so that you can select the path for the HTML Exports. The parameter should be the empty string ("").

Examples:

External("POP3-DlgHTMLExportPath", "")

See Also:

POP3-AttachPath (37)
POP3-DlgAttachPath (37)
POP3-HTMLExportPath (38)

POP3-Connect

This function opens a connection to your POP3 mail server using your username and password to log in. The parameter should be the empty string (""). Note: In previous versions of POP3it, this function was named POP3-OpenConnection. You can still use POP3-OpenConnection, but it simply calls this function.

Example:

External("POP3-Connect", "")

See Also:

POP3-Host (35)
POP3-Username (36)
POP3-Password (36)
POP3-APOP (36)
POP3-Disconnect (39)

POP3-Disconnect

This function closes the connection to your POP3 mail server. The parameter should be the empty string (""). Note: In previous versions of POP3it, this function was named POP3-CloseConnection. You can still use POP3-CloseConnection, but it simply calls this function.

Example:
External("POP3-Disconnect", "")

See Also:
POP3-Connect (38)

POP3-AcquireFirst

The POP3-AcquireFirst function sets up the first email for download and returns the number of emails on the server available for download. You must be connected to the mail server for this function to work. If there are one or more emails available for download then you can use the "Current" functions below to retrieve information for the first email. If there are no emails available for download, then this function will return zero ("0"). The parameter should be the empty string (""). Note: If the need arises, power users can start downloading at a specific place or download specific emails by number by using the number of the email as the parameter.

Example:
External("POP3-AcquireFirst", "")

See Also:
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrUniqueID

This function returns the Unique Identifier of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The UniqueID of an email is useful if you leave your email on the server instead of deleting it so that you can download it with an alternate email client. You can use the Unique Identifier to determine if you have already downloaded this email into your database. See the POP3it Example database and the POP3it Example Explained section earlier in this documentation for an example of how to use the UniqueID. The parameter should be the empty string ("").

Example:
External("POP3-CurrUniqueID", "")

See Also:
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrSize

This function returns the size of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The parameter should be the empty string. Note: If there is an attachment in the email, the size of the email is usually 33% larger than the actual size of the attachment because of encoding.

Example:

External("POP3-CurrSize", "")

See Also:

POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrHeader

This function will return the entire "Headers" section of the "Current" email if the parameter is the empty string (""). However, if you only need a specific header out of the email, then specify the name of the header as the parameter. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work.

Examples:

External("POP3-CurrHeader", "")
External("POP3-CurrHeader", "CC")
External("POP3-CurrHeader", "X-Priority")

See Also:

POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrDate

This function returns the Date of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. This function will force the date to conform to the RFC822 internet standard. This allows you to create a simple calculation to turn it into a FileMaker Pro Date or Time field. This function will return the Date in this form: "Mon, 06 Aug 2001 03:50:33 -0500" For an example calculation that turns an email date into a FileMaker Pro Date, please see the POP3it Example database and the POP3it Example Explained section earlier in this documentation. The parameter should be the empty string ("").

Example:

External("POP3-CurrDate", "")

See Also:
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrTo

This function returns all To email addresses from the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The parameter should be the empty string ("").

Example:
External("POP3-CurrTo", "")

See Also:
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrFrom

The POP3-CurrFrom function returns the From email address of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The parameter should be the empty string ("").

Example:
External("POP3-CurrFrom", "")

See Also:
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrSubject

This function returns the Subject of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The parameter should be the empty string ("").

Example:
External("POP3-CurrSubject", "")

See Also:
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrBodyType

This function will tell you the type of Body in the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. This function will return "TEXT" if the email only has a plain text part, "HTML" if the email only has an HTML part, or "BOTH" if the email is "multipart/alternative" and has both Text and HTML parts. The parameter should be the empty string ("").

Example:

External("POP3-CurrBodyType", "")

See Also:

POP3-AcquireFirst (39)
POP3-CurrBody (42)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrBody

This function returns the "Body" or "Content" of the "Current" email. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. You can specify what type of Body POP3it should return to you. If you only want Plain Text, then use the parameter "Text". If you only want HTML, use the parameter "HTML". For "multipart/alternative" emails that have both Text and HTML parts, you can use the parameter "Both" to return the Plain Text part followed by the HTML part. For single part emails, "Both" will return which ever type is available. There are also two other parameters you can use: "Text, HTML" and "HTML, Text". The first will return a Plain Text body if there is one, otherwise it will return an HTML part. The second will return an HTML part, unless there is not one, and then it will return a plain Text part. If you do specify the empty string ("") as the parameter, POP3it uses the default body type in the Configuration Dialog.

Examples:

External("POP3-CurrBody", "")
External("POP3-CurrBody", "Text")
External("POP3-CurrBody", "HTML")
External("POP3-CurrBody", "Both")
External("POP3-CurrBody", "Text, HTML")
External("POP3-CurrBody", "HTML, Text")

See Also:

POP3-AcquireFirst (39)
POP3-CurrBodyType (42)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrAttach

The POP3-CurrAttach function will decode and save any attachments in the "Current" email to the location on your hard drive that you specified with the POP3-AttachPath or POP3-DlgAttachPath functions or in the Configuration Dialog. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. This function will return the path(s) to the file(s) on your hard drive where the attachments were saved. The parameter should be the empty string ("").

Example:

External("POP3-CurrAttach", "")

See Also:

POP3-AttachPath (37)
POP3-DlgAttachPath (37)
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrHTMLExport

The POP3-CurrHTMLExport function will export the HTML part of an email to a file, complete with any inline images, into the HTML Export directory that you have specified with the POP3-HTMLExportPath or POP3-DlgHTMLExportPath functions or in the Configuration Dialog. You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. If the current email does not contain an HTML part, this function will do nothing. This function will return the path to the HTML file on your hard drive where the HTML part was saved. You must specify a file name for the HTML file as the parameter.

Examples:

External("POP3-CurrHTMLExport", "MyHTMLExportFile")
External("POP3-CurrHTMLExport", Subject)

See Also:

POP3-HTMLExportPath (38)
POP3-DlgHTMLExportPath (38)
POP3-AcquireFirst (39)
All "Current" functions
POP3-AcquireNext (44)

POP3-CurrAll

This function returns the entire "Current" email, including the entire "Headers" section of the email and the "Body" or "Content" of the email. It will also decode and save all attachments into your attachment directory and return the path(s) to those file(s). You must be connected to your mail server, and there must be at least one email on the mail server for this function to work. The parameter to this function is exactly like the parameter in the POP3-CurrBody function.

Examples:

External("POP3-CurrAll", "")
External("POP3-CurrAll", "Text")
External("POP3-CurrAll", "HTML")
External("POP3-CurrAll", "Both")
External("POP3-CurrAll", "Text, HTML")
External("POP3-CurrAll", "HTML, Text")

See Also:

POP3-AcquireFirst (39)
POP3-CurrBodyType (42)
POP3-CurrBody (42)
All "Current" functions
POP3-AcquireNext (44)

POP3-AcquireNext

The POP3-AcquireNext function sets up the next email for you to use with the "Current" functions. You must be connected to the mail server for this function to work. This function will return the "Current email number starting with the second email, which is where POP3-AcquireFirst left off, and report each email number until no emails are left for download. When there are no emails left, POP3-AcquireNext will return zero ("0"). This gives you a test to break out of your email downloading loop. The second to last step in your loop will call this function, then the last step in your loop will test if this function returned "0". If it did not, then it can continue the loop, otherwise it will break out. The parameter should be the empty string ("").

Example:

External("POP3-AcquireNext", "")

See Also:

POP3-AcquireFirst (39)
All "Current" functions

POP3-CurrDelete

This function marks the "Current" email for deletion. You must be connected to the mail server, and there must be at least one email on the mail server for this function to work. If the POP3 connection is closed properly, then the mail server will delete all marked emails. The parameter should be the empty string ("").

Example:

External("POP3-CurrDelete", "")

See Also:

POP3-DeleteAll (45)
POP3-UnDeleteAll (45)

POP3-DeleteAll

The POP3-DeleteAll function will mark every email on the mail server for deletion. You must be connected to the mail server for this function to work. If the POP3 connection is closed properly, then the mail server will delete all marked emails. The parameter should be the empty string ("").

Example:

External("POP3-DeleteAll", "")

See Also:

POP3-CurrDelete (45)
POP3-UnDeleteAll (45)

POP3-UnDeleteAll

The POP3-UnDeleteAll function clears all deletion marks on all emails on the mail server. You must be connected to the mail server for this function to work. When the POP3 connection is closed, no emails will be deleted unless you use POP3-CurrDelete or POP3-DeleteAll again. The parameter should be the empty string ("").

Example:

External("POP3-UnDeleteAll", "")

See Also:

POP3-CurrDelete (45)
POP3-DeleteAll (45)

POP3-AutoDelete

This function allows you to tell POP3it to automatically mark your email for deletion for you so that you do not have to call POP3-CurrDelete or POP3-DeleteAll. If the parameter you give is the empty string (""), POP3it will use the default setting you have in the Configuration Dialog. If the parameter is "Never", POP3it will never automatically mark your mail for deletion. If the parameter is "Before AcquireNext", POP3it will mark each individual email for deletion before it gets the next email. If the parameter is "Before Disconnect", POP3it will mark all email for deletion before it closes the connection. Example use:

Examples:

```
External("POP3-AutoDelete", "")
External("POP3-AutoDelete", "Never")
External("POP3-AutoDelete", "Before AcquireNext")
External("POP3-AutoDelete", "Before Disconnect")
```

See Also:

POP3-CurrDelete (45)
POP3-DeleteAll (45)
POP3-UnDeleteAll (45)

POP3-AutoCheckEmail *

Use this function to tell POP3it to automatically check email every so many minutes. POP3it does this by calling a script in your database which starts the transaction. The parameter consists of the database the script is in, the name of the script to call, and the interval in minutes, all separated by the pipe character ("|"). ("|" is created by typing shift-backslash.) You must specify the full name of the database, including the ".fp3" or ".fp5" extension. (Macintosh users may not have the extension, depending on how you originally named the file.) The first example will call the script "Auto Check Mail" in the database "POP3it_Example.fp3" every 15 minutes.

Examples:

```
External("POP3-AutoCheckEmail", "POP3it_Example.fp3|Auto Check Mail|15")
External("POP3-AutoCheckEmail", Status(CurrentFileName) & "|Check POP3 Account|"
        & AutoCheckInterval)
```

Obsolete Functions

POP3-OpenConnection

Renamed in version 3.0. To help our users migrate from one of our plug-ins to the next with ease, we renamed the POP3-OpenConnection function to POP3-Connect to conform with the names in our other plug-ins. POP3-OpenConnection will still work if you have an old database that worked with a previous version of POP3it, but it will simply call the POP3-Connect function.

POP3-CloseConnection

Renamed in version 3.0. To help our users migrate from one of our plug-ins to the next with ease, we renamed the POP3-CloseConnection function to POP3-Disconnect to conform with the names in our other plug-ins. POP3-CloseConnection will still work if you have an old database that worked with a previous version of POP3it, but it will simply call the POP3-Disconnect function.

POP3-CurrHead

Removed in version 3.0. In previous versions of POP3it, there was some confusion between the POP3-CurrHead function and the POP3-CurrHeader Function. The former returned the entire "Headers" section of the email, while the later returned only the specific header you asked for. These functions have been combined into one function named POP3-CurrHeader. Please see the POP3-CurrHeader function above for a description of how it works. You can still call this function, but it will simply call the POP3-CurrHeader function.

Understanding Error Responses

Every function in POP3it returns a response indicating the success or failure of that function. If the function is successful, it will return a response indicating that it set the value you were trying to set, or completed the task that needed to be completed. If however, the function is not successful, it will return an Error Response. This Error Response is in the form of:

ERROR: <Function Name>: <Error Description>

For example, if you forgot to set a POP3 Host using POP3-Host, and you attempt to connect to your mail server with POP3-Connect, the POP3-Connect function will return the following Error Response:

ERROR: Connect: Could not connect: No Host specified.

Error responses always start with the word "ERROR" in all caps, followed by a colon, followed by the function that had the error, followed by a colon, followed by the actual error that occurred. You can use the various FileMaker Pro Text Functions to extract the different parts of the Error Response for your own use. For instance, if you want to know if the response you just got back was an ERROR, you can use the LeftWords function to return the first word and see if it is an error.

Example:

```
Set Field ["POP3it Result", "External("POP3-Connect", "")"]
If ["LeftWords(POP3it Result, 1) = "ERROR""]
  <inform the user and exit>
Else
  <download the emails>
End If
```

Credits

Programming, documentation, and example databases by Jake Traynham
Concept, web design, and example databases by Jesse Traynham

Contact Information

Email:	POP3it@comm-unity.net
POP3it Website:	http://www.pop3it.com/
Main Website:	http://www.cnsplug-ins.com/
Phone:	817-560-4226

You can write us at

Comm-Unity Networking Systems
8652 Hwy 80 West
Fort Worth, Texas 76116