



Address Book@ **Manipulator**

Developer's Guide

Table of Contents

I. Introduction	3
II. Integration Steps	4
1) Installing the Plug-in	4
2) Talking to Address Book.....	5
<i>A. Pushing to Address Book</i>	<i>5</i>
<i>B. Pulling from Address Book.....</i>	<i>7</i>
<i>C. Searching Address Book Records</i>	<i>9</i>
3) Handling Groups.....	10
III. Contact us.....	11

I. Introduction

Description:

The Address Book Manipulator plug-in offers functions that support a bi-directional data exchange between FileMaker® and Apple® Address Book. With this plug-in FileMaker users are able to 'push' new records into Address Book, 'pull' records from Address Book, and modify existing records in Address Book. These operations are accomplished by using FileMaker function calls from within FileMaker calculations. These calculations are generally determined from within FileMaker 'SetField' or 'If' script steps.

Intended Audience:

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

Successful Integration Practices:

- 1) Read the Developer's Guide
- 2) Read the Functions Guides
- 3) Look at our FileMaker Demo and video tutorials

Demo and video tutorials can be found here: <http://www.addressbookmanipulator.com>

- 4) Familiarize yourself with Apple's Address Book

II. Integration Steps

Accessing and using the plug-in involves the following steps:

1) Installing the Plug-in

Installing the Plug-in:

The first step is to install the plug-in into FileMaker.

- 1) Quit FileMaker Pro completely.
- 2) Locate the plug-in in your download which will be located in a folder called "Plug-in". On Mac the plug-in will have a ".fmplugin" extension.
- 3) Copy the actual plug-in and paste it to the Extensions folder which is inside the FileMaker program folder. On Mac this is normally located here: Volume/Applications/FileMaker X/Extensions (Volume is the name of the mounted volume)
- 4) Start FileMaker. Confirm that the plug-in has been successfully installed by navigating to "Preferences" in FileMaker Pro, then click the "Plug-ins" Tab. There you should see the plug-in listed with a corresponding check box. This indicates that you have successfully installed the plug-in

Registration:

The next step is to register the plug-in which enables all plug-in functions.

- 5) Confirm that you have access to the internet and open our FileMaker demo file, which can be found the in "FileMaker Demo File" folder in your original download.
- 6) If you are registering the plug-in in Demo mode, then simply click the "Register the Plug-in" button and do not change any of the fields. Your plug-in should now be running in "DEMO" mode. The mode is noted in the upper right hand corner of our FileMaker Demo file on the Setup tab.
- 7) If you are registering a licensed copy, then simply enter your license number in the "LicenseID" field and click the "Register the Plug-in" button. Make sure you remove the Demo License ID and enter your registration information exactly as it appears in your confirmation email. Your plug-in should now be running in "LIVE" mode. The mode is noted in the upper right hand corner of our FileMaker Demo file on the Setup tab.

Congratulations! You have now successfully installed and registered the plug-in!

Why do I need to Register?

In an effort to reduce software piracy, Productive Computing, Inc. has implemented a registration process for all plug-ins. The registration process sends information over the internet to a server managed by Productive Computing, Inc. The server uses this information to confirm that there is a valid license available and identifies the machine. If there is a license available, then the plug-receives an acknowledgment from the server and installs a certificate on the machine. This certificate never expires. If the certificate is ever moved, modified or deleted, then the client will be required to register again. On Mac this certificate is in the form of a plist file.

The registration process also offers developers the ability to automatically register each client machine behind the scenes by hard coding the license ID in the PCAB_Register function. This proves beneficial by eliminating the need to manually enter the registration number on each client machine. There are other various functions available such as PCAB_GetOperatingMode and PCAB_Version which can assist you when developing an installation and registration process in your FileMaker solution.

2) Talking to Address Book

Moving information between FileMaker and Apple's Address Book involves either pushing or pulling records from one application into the other. Using the plug-in you are able to push information from FileMaker to Address Book or pull information from Address Book to FileMaker.

Pushing or pulling typically involves first opening an Address Book record, then reading or writing to or from that record. The Address Book ID is used to identify if the record already exists. This can be useful in determining if the contact is a new contact or an existing contact to be edited or deleted. If the contact does not yet exist in Address Book, then there is not an existing Address Book record to first open and you can simply add a new record.

This may be better explained with an example. Let's have a closer look at pushing and pulling records.

A. Pushing to Address Book

Pushing information to Address Book is the same as making or maintaining a copy of a FileMaker record in Address Book. This is accomplished by first creating a blank record in AddressBook, or opening an existing record, then setting the values for the different properties (or fields), and finally saving the new AddressBook record.

A list of all the functions necessary to create and populate an AddressBook record follows:

PCAB_New(Type)

PCAB_Open(AddressBookID)

PCAB_SetValueForProperty(Property ; Value)

PCAB_AddAddress(Label ; Street ; City ; State ; Zip ; Country ; CountryCode)

PCAB_AddMV(Property ; Value ; Label)

PCAB_Save

Each of the above functions is described in the Functions Guide that accompanies this document. This guide lists the meaning and allowable values for the parameters of each of the functions. For the sake of brevity we will cover only the basics of each function.

PCAB_New(Type) creates a new empty record in Address Book. There are two available types of records to create, a "Contact" type or a "Group" type. This function returns Address Book's unique identifier for the newly created record.

PCAB_Open(AddressBookID) opens an existing record in Address Book for editing or reading. The Address Book must be available to the FileMaker solution before a record can be accessed using this function.

PCAB_SetValueForProperty(Property ; Value) is used to set the value for any of the Address Book record's single valued properties. A single valued property is one that can have only one value in Address Book, such as a "First Name" or "Last Name".

PCAB_AddMV(Property ; Value ; Label) is used to add a value to a multi valued property. Multi valued properties are those in Address Book that may have more than one value, such as a phone number or an e-mail address.

PCAB_AddAddress(...) is pretty straight forward. It simply adds an address to the Address Book record with the values specified. An address is a multi valued property so several addresses can be added to any Person record in Address Book.

PCAB_Save saves the record and its contents to the Address Book.

Always remember to save any changes after you are finished.

An example script adding a contact's name, address, and phone number follows:

```
SetField[ Contacts::ABID ; PCAB_New( "Contact" ) ]
If [ LeftWords(AddressBookID ; 1 ) <> "!!ERROR!!" ]
SetField[ gResult ; PCAB_SetValueForProperty( "First Name" ; table::NameFirst ) ]
SetField[ gResult ; PCAB_SetValueForProperty( "Last Name" ; table::NameLast ) ]
SetField[ Contacts::addr1UID ; PCAB_AddAddress( "work";"1 St.";
"AnyTown"; "AnyState"; "11111"; "USA"; "us" ) ]
SetField[ Contacts::phone1UID ; PCAB_AddMV( "Phones" ; table::workPhone ; "work" ) ]
SetField[ gResult ; PCAB_Save ]
End if
```

B. Pulling from Address Book

Once a record is opened using either `PCAB_Open`, `PCAB_GetFirstRecord`, or `PCAB_GetNextRecord` the opened record is available for reading. Pulling the values of the different properties/fields may require the use of a single function or a combination of functions depending on the type of property being accessed.

Properties of Address Book Records are divided into two types: Single Valued properties and Multi Valued properties. Single Valued properties are very easy to access and require only one function, the `GetValueForProperty` function.

`PCAB_GetValueForProperty(Property)` will return the value stored in the single valued property identified by the 'Property' parameter. See the accompanying Functions Guide for valid 'Property' values.

Multi valued properties require a bit more work to get at the data. This is because Address Book holds a unique identifier for each value stored in the multi valued property. Accessing the value (and its label) stored in a mv property requires knowing what the identifier is for the particular value you desire. Gathering these identifiers is accomplished with the two functions `OpenFirstMVProperty` and `OpenNextMVProperty`.

`PCAB_OpenFirstMVProperty(Property)` returns the identifier for the first value stored in the property identified by 'Property.' See the Functions Guide for valid property values.

`PCAB_OpenNextMVProperty` returns the identifier for the next value stored in the property of the active property. The active property is set with the `PCAB_OpenFirstMVProperty` function.

Once the identifier for all values stored in a property are stored, then the developer can access the values and labels stored in that property. This is done with the `GetLabelForUID` and `GetValueForUID` functions.

`PCAB_GetLabelForUID(Property ; UID)` returns the label (home , work, etc..) for the value identified by the 'UID' parameter. 'Property' is the name of the property that is holding the UID.

`PCAB_GetValueForUID(Property ; UID ; optKey)` returns the value identified by the UID. 'Property' is the property holding the UID. 'optKey' is optional and is used only when the property is an Address.

An example script pulling contact information follows:

```
#perform a search
...
#open the record
SetField[ Contacts::ABID ; PCAB_OpenFirstRecord ]
#
#get the UIDs for the 2 addresses and 2 phones
SetField[ Contacts::Addr1UID ; PCAB_OpenFirstMVProperty( "Address" ) ]
SetField[ Contacts::Addr2UID ; PCAB_OpenNextMVProperty ]
SetField[ Contacts::ph1UID ; PCAB_OpenFirstMVProperty( "Phones" ) ]
SetField[ Contacts::ph2UID ; PCAB_OpenNextMVProperty ]
#
#get a couple single valued properties
SetField[ Contacts::NameFirst ; PCAB_GetValueForProperty( "First Name" ) ]
SetField[ Contacts::NameLast ; PCAB_GetValueForProperty( "Last Name" ) ]
#
#get first Address
SetField[ Contacts::addrLabel1 ; PCAB_GetLabelForUID( "Address" ; Contacts ::Addr1UID ) ]
SetField[ Contacts ::Street1 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr1UID ; "Street" ) ]
SetField[ Contacts::City1 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr1UID ; "City" ) ]
SetField[ Contacts::State1 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr1UID ; "State" ) ]
SetField[ Contacts::ZIP1 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr1UID ; "Zip" ) ]
#
#get second address
SetField[ Contacts::addrLabel2 ; PCAB_GetLabelForUID( "Address" ; Contacts ::Addr2UID ) ]
SetField[ Contacts::Street2 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr2UID ; "Street" ) ]
SetField[ Contacts::City2 ; PCAB_GetValueForUID( "Address" ; Contacts ::Addr2UID ; "City" ) ]
#
#get first phone
SetField[ Contacts::phLabel1 ; PCAB_GetLabelForUID( "Phones" ; Contacts ::ph1UID ) ]
SetField[ Contacts::phone1 ; PCAB_GetValueForUID( "Phones" ; Contacts ::ph1UID ) ]
#
#get second phone
SetField[ Contacts::phLabel2 ; PCAB_GetLabelForUID( "Phones" ; Contacts ::ph2UID ) ]
SetField[ Contacts::phone2 ; PCAB_GetValueForUID( "Phones" ; Contacts ::ph2UID ) ]
#
#all done
#
```

C. Searching Address Book Records

When pulling from Address Book the developer needs to first open a record in Address Book before its fields/properties can be accessed. There are two different ways to open Address Book records, directly using PCAB_Open, or indirectly using PCAB_Search together with PCAB_OpenFirstRecord and PCAB_OpenNextRecord. The PCAB_Search function is a powerful searching tool offered by the plug-in. It easily locates records that contain certain property/field values. It is a simple and efficient tool for finding records in the Address Book. On success the PCAB_Search will return the number of records meeting the search criteria. This value returned is then used to determine if any records are in the found set. If there are records in the found set then you can begin iteration through the found set, opening each record for reading and writing.

An example follows:

```
SetField[ gResult ; PCAB_Search("Person" ; "Modified Date" ; "" ; "" ; 3600 ; "WithinIntervalAroundToday" ) ;  
if[gResult > 0 ]  
SetField[ gResult ; PCAB_OpenFirstRecord ]  
Loop  
Exit Loop If [LeftWords( gResult ; 1 )="!!ERROR!!" or LeftWords( gResult ; 1 )="END" ] ]  
#script steps to pull property/field values  
...  
#done with getting fields, so open next record  
SetField[ gResult ; PCAB_OpenNextRecord ]  
End Loop  
...
```

3) Handling Groups

You can create a new group by using the PCAB_New(Type) function whereas the type specifies the type of record to create. The type should be "Group". You can also create a new subgroup by using the PCAB_AddSubGroup(Address Book ID).

When handling groups, the script needs to open the group record and then add contacts to the specified group. To open the Group record you will need to locate the group using the PCAB_Search function. For example, PCAB_Search("Group" ; "Name" ; "" ; "" ; "Sales" ; "Equal"). This will find the group with the name "Sales". The search will return the number of records found that match the search criteria. The scripts use the PCAB_OpenFirstRecord function to get the Address Book ID of the record. Then with the Group record opened use the PCAB_AddPerson function to add contacts to the group. You have now successfully added a contact to a group.

Since we do not yet have a FileMaker demo of how to add a new contact to an existing group, please allow me to provide you with the general steps with a practical example below:

1) Locate and open the desired group using the PCAB_Search function. In this example our group name will be called "Sales"

PCAB_Search("Group" ; "Name" ; "" ; "" ; "Sales" ; "Equal"). The search will return the number of records found that match the search criteria.

2) Open the desired group record using PCAB_OpenFirstRecord. This function opens the first record in the found set from the previous call to PCAB_Search.

3) Add new contact record to the Sales group by using PCAB_AddPerson.
The new contact record will automatically be added to the currently opened group.

4) Save the record using PCAB_Save

The script will look something like this:

```
PCAB_Search( "Group" ; "Name" ; "" ; "" ; "Sales" ; "Equal" )
PCAB_OpenFirstRecord
PCAB_AddPerson
PCAB_Save
```

III. Contact us

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

Phone: 760-510-1200

Email: support@productivecomputing.com

Forum: www.productivecomputing.com/forum

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