Productive Computing

Biometric Fingerprint Reader

Functions Guide

Revised February 26, 2019

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

Table of Contents

I. Introduction

Description

The Biometric Fingerprint Reader plug-in from Productive Computing offers functions that allow you to incorporate biometric fingerprint authentication, security and script control options in FileMaker Pro. The plug-in is intended to be used with Digital Persona U.areU 4500 Reader device. With this plug-in technology, FileMaker Pro can recognize and authenticate individuals based on who they are, instead of what they know (passwords and PINs) or what they posses (keys and swipe cards). 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 Guide
3) Review our FileMaker Demo: http://www.productivecomputing.com/dl/Biometric/Biometric_Reader.zip
4) Watch video tutorials: http://www.productivecomputing.com/video/?p=1236

II. Function Descriptions

This section describes the functions that are available with the Biometric Fingerprint Reader plug-in.

All functions, except where noted, return a text string. It is up to the user to convert the returned value when necessary.

A. Error Handling and Registration Related Functions

PCFP_GetLastError(type)

Purpose:
Returns a textual or numerical description of the last error encountered by the plug-in corresponding to the previous function call. This function gives the developer the option to trap errors as a text or number.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
type	Determines which type of error value is returned, text or number.	"text" or "number"	"text"

Return Values:
An error description or error number depending on the provided parameter, otherwise an !!ERROR!!

Example:
PCFP_GetLastError("text")

PCFP_GetOperatingMode

Purpose:
Returns a string representing the current operating mode of the plug-in.

Dependencies:
None

Parameters:

Return Values:
"LIVE", "DEMO", "UNREGISTERED", "EXPIRED" or otherwise an !!ERROR!!

Example:
PCFP_GetOperatingMode

PCFP_Register(ServerName ; ServerPort ; ServerPage ; LicenseID)

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

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
ServerName	The server of the Productive Computing registration server.	licensing.productivecomputing.com	None
ServerPort	The port on which the server can be accessed.	80	None
ServerPage	The page at which the registration code is accessed.	/PCIReg/pcireg.php	None
LicenseID	The license id received with purchase of the plug-in or DEMO-PCFP.	"Your license ID" or "DEMO-PCFP"	None

Return Values:
Returns 0 on success, otherwise an !!ERROR!! or error text. Anything other than a 0 is an error.

Example:
PCFP_Register("licensing.productivecomputing.com" ; "80" ; "/PCIReg/pcireg.php" ; "DEMO-PCFP")

PCFP_Version(type)

Purpose:
Used to identify the plug-in name and plug-in version installed on a machine.

Dependencies: None

Parameters:

Parameter Name	Purpose	Values	Default Value
type	Determines the version string to be returned. The "short" version string includes only the version number of the plug-in. Example: "1.0.1.0". The "long" includes the name of the plug-in and version number. Example: "Biometric Fingerprint Reader 1.0.1.0"	"long" or "short"	"short"

Return Values:
A short or long version string, otherwise an !!ERROR!!

Example:
PCFP_Version("long")

PCFP_AddFingerprintToUser(UserID ; FingerprintData ; FingerPosition)

Purpose:
This function adds a fingerprint template to the DigitalPersona manager. If a user does not exist under the provided "UserID", then a new user is created with that id.

Dependencies:
Device manager must be initialized.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserID	The id of the user you wish to add fingerprint data to, or a new user id you wish to create.		none
FingerprintData	A fingerprint data template file.		none
FingerPosition	The finger number of the corresponding fingerprint data	1-10	none

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

Example:
PCFP_AddFingerprintToUser("Taylor"; "Fingerprint Template File"; "3")

PCFP_DeleteAllUsers

Purpose:
This function removes all users and their fingerprints from the device manager.

Dependencies:
Device manager must be initialized.

Parameters:

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

Example:
PCFP_DeleteAllUsers

PCFP_DeleteUser(UserID)

Purpose:
Deletes a specific user and their fingerprints from the device manager.

Dependencies:
Device manager must be initialized and provided user must exist in the device manager.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserID	Identifies the user to be deleted from the manager.	Any existing userID name.	None

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

Example:
PCFP_DeleteUser("Sally")

PCFP_EnrollUser(UserID)

Purpose:
This function adds the new user identified by the UserID parameter to the device manager and stores their information under the provided user id.

Dependencies:
Device manager must be initialized and there must be memory remaining from the initially allocated space.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserID	The user ID name to identify the newly enrolled user.		None

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

Example:
PCFP_EnrollUser("Sally")

PCFP_GetFingersForUser(UserID)

Purpose:
Retrieves the string of number(s) representing each finger enrolled for a given user id. For example, 1 represents the right thumb, 2 represents the right index finger, 3 represents the right middle finger, 4 represents the right ring finger, 5 represents the right pinky, 6 represents the left thumb, 7 represents the left index finger, 8 represents the left middle finger, 9 represents the left index finger and 10 represents the left pinky.

Dependencies:
Device manager must be initialized and user must exist in system.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserID	Identifies the user to return finger position data for.		None

Return Values:
A space separated string of numbers (1-10) representing the fingers for which the user has fingerprints stored, otherwise an !!ERROR!!

Example:
PCFP_GetFingersForUser("Sally")

PCFP_Identify(optPrintCount)

Purpose:
Identifies a user based upon their provided fingerprints.

Dependencies:
Device manager must be initialized and users must exist in the system.

Parameters:

Parameter Name	Purpose	Values	Default Value
optPrintCount	Determines how many fingerprints are required to identify a user.	1-10	1

Return Values:
Returns UserID on success, otherwise an !!ERROR!!

Example:
PCFP_Identify("3")

PCFP_ImportUserFromDevice(UserID ; FingerPosition)

Purpose:
This function retrieves a fingerprint template corresponding to the provided UserID and FingerPosition.

Dependencies:
Device manager must be initialized and user must exist on device with corresponding fingerprint.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserID	The user to retrieve fingerprint data for.		None
FingerPosition	The specific finger to retrieve data for.	1-10	None

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

Example:
PCFP_ImportUserFromDevice("Sally"; "2")

PCFP_Initialize(MaxFingerprints)

Purpose:
This function initializes the Digital Persona device manager and allows you to specify how many fingerprints will be allowed to be stored in the device manager.

Dependencies:
None

Parameters:

Parameter Name	Purpose	Values	Default Value
MaxFingerprints	The number of fingerprints you are able to store in the device manager at a given time.	Any positive number	none

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

Example:
PCFP_Initialize("500")

PCFP_GetFingerprintImageForUser(UserID ; FingerPosition)

Purpose:
This function returns the fingerprint image for the specified user and the provided finger position, if it exists. The data returned by the function is binary picture data and should be stored directly into a container field to display the image.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserName	The name or ID of the user		
FingerPosition	The finger number of the corresponding fingerprint image	1-10	

Return Values:
Binary data of the fingerprint image on success, or "!!ERROR!!"

Notes:
Fingerprints that were enrolled with version 1 of the Biometric Fingerprint Reader plug-in will not have a corresponding fingerprint image for them. Only fingers enrolled with version 2 and newer will be able to get fingerprint images.

PCFP_AddFingerprintImageToUser(UserID ; FingerImageData ; FingerPosition)

Purpose:
This function accepts a binary fingerprint image data file and stores it into the Biometric DLL file, associating the image with the user for the finger position provided. This finger image can later be retrieved using PCFP_GetFingerprintImageForUser.

Parameters:

Parameter Name	Purpose	Values	Default Value
UserName	The name or ID of the user		
FingerImageData	Binary image data of the fingerprint		
FingerPosition	The finger number of the cooresponding fingerprint image	1-10	

Return Values:
0 on success, or "!!ERROR!!"

Notes:
Fingerprints that were enrolled with version 1 of the Biometric Fingerprint Reader plug-in will not have a corresponding fingerprint image for them. Only fingers enrolled with version 2 and newer will be able to get fingerprint images.

PCFP_ListReaders

Purpose:
This function generates a return-separated list of compatible fingerprint reader devices connected to the current machine. Each entry is a unique identifier ("ID") for the reader.

Parameters:
None.

Return Values:
A list of compatible fingerprint readers, or "!!ERROR!!" for an error.

PCFP_SelectReader(ReaderID)

Purpose:
This function instructs the plug-in to use the reader specified by the ReaderID as the primary fingerprint reader for fingerprint scanning.

Parameters:

Parameter Name	Purpose	Values
ReaderID	Unique identifier for the reader	

Return Values:
0 on success, or "!!ERROR!!"

PCFP_GetReaderProperty(ReaderID ; PropertyName)

Purpose:
This function returns the value of a property of the specified reader. This is useful for determining version information, firmware data, fingerprint scanner type and other features about the reader.

Parameters:

Parameter Name	Purpose	Values
ReaderID	Unique identifier for the reader	
PropertyName	Name of the property	

Return Values:
0 on success, or "!!ERROR!!"

Notes:
If the ReaderID parameter is blank, the property will be pulled from the current active reader.

PCFP_GetReaderCapabilities (ReaderID)

Purpose:
This function returns a return-separated list of reader capabilities. Capabilities are features that the fingerprint scanner device can perform.
Parameters:

Parameter Name	Purpose	Values
ReaderID	Unique identifier for the reader	

Return Values:
A list of capabilities on success, or "!!ERROR!!"

Notes:
The capabilities are specifically those supported by the device itself. The plug-in's SDK is also able to identify, match and control capture, even if the reader does not state that it can perform those actions.

PCFP_CalibrateReader(ReaderID)

Purpose:
This function instructs the reader to perform a calibration test, recalibrating its sensors. This is useful if the reader is experiencing some issues with
Parameters:

Parameter Name	Purpose	Values
ReaderID	Unique identifier for the reader	

Return Values:
0 on success, or "!!ERROR!!"

Notes:
Only readers that have the "CALIBRATE" capability will be able to calibrate.

Reader Properties

Property Name	Definition/Meaning
ID	Unique identifier of the reader device
Name	Descriptive name of the reader device
Modality	Modality (detection type). "Unknown", "Swipe" or "Area".
Status	Reader's status. "Busy", "Failure", "Needs Calibration", "Reader" or "Unknown".
Technology	Reader type. "Capacitive", "Optical", "Pressure", "Thermal" or "Unknown".
FirmwareVersion	Version of the device's firmware.
HardwareVersion	Version of the device's hardware.
ProductName	Descriptive name of the device
VendorName	Name of the device's manufacturer

Reader Capabilities

Capability Name	Definition/Meaning
CAPTURE	The ability to capture fingerprints
IDENTIFY	The ability to perform on-machine identification
MATCH	The ability to perform fingerprint matching
STREAM	The ability to stream fingerprint scanning images
EXTRACT_FEATURES	The ability to extract fingerprint feature data from a scanned fingerprint
CALIBRATE	The ability to perform self-calibration of the fingerprint sensors
FINGERRINT_STORAGE	The ability to store fingerprint information directly on the device
POWER_MANAGEMENT	The ability to manage device power requirements
PIV_COMPLIANT	Indicator that the device is PIV compliant.

III. Return/Error Codes

When something unexpected happens, a plug-in function will return a result of !!ERROR!!. This makes it simple to check for errors. If a plug-in function returns !!ERROR!!, then immediately after call PCFP_GetLastError("Text") function for a detailed description of what the exact error was or PCFP_GetLastError("Number") for an error number.

We find that most developers run into issues due to a lack of error trapping. Please ensure that you properly trap for errors in your solutions. Here are a few samples of how you can check for errors.

```
Set Variable [ $result = MyPluginFunction( "a" ; "b" ; "c" ) ]
If [ $result = !!ERROR!! ]
Show Custom Dialog [ "An error occurred: " & PCFP_GetLastError( "Text" ) ]
End If
```

The PCFP_GetLastError(format) function gives you the option to display the error description or error number. Displaying the error number is more user friendly in international environments, where an English error description may not be desired. If the format parameter is set to "Number" such as PCFP_GetLastError("Number"), then an error number will be returned. If format parameter is empty such as PCFP_GetLastError or PCFP_GetLastError("Text"), then an English error description will be returned. The error numbers and their meanings can be found below.

Value	Meaning
0	Success
-1	Plug-in not registered or session expired
-3	Invalid # of Parameters
-4	Invalid Parameter value(s)
-10	Failed Registration
-9001	Process timeout
-9002	No active passport
-9003	User does not exist in system
-9004	Unable to create id from name
-9005	SDK Error:
-9006	App Data Error:
-9007	System error:
-9008	User has fingerprint at that position
-9009	This template already exists in the database
-9012	That user already exists
-9013	You must initialize
-9014	User canceled enrollment
-9015	An instance of the UAREUManager.dll was not created

IV. Contact Us

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

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

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