



Developer's Guide

Revised July 21, 2011

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

© Copyright 2011 Productive Computing, Inc.

Table of Contents

I. Introduction.....	3
II. Integration Steps.....	4
1) Installing and Registering the Plug-in	4
2) Changing Printers.....	6
3) Get Valid Printer Names	7
4) Automatic Printing	8
5) Automatic Printer Selection Without Printing.....	9
6) Optional Printing Parameters or Attributes.....	9
7) Get or Set System Printer	10
III. Error Handling	11
IV. Contact Us	12

I. Introduction

Description

The Change Printer plug-in is a simple tool designed to aid in the task of printing records in a FileMaker® Pro database. The plug-in allows for the FileMaker user to design scripts that will dynamically change printing from one printer to another allowing you to set a series of printing attributes. The plug-in also has tools that can identify all printers currently available to a given user and ways to get and set the operating system default printer. These operations are accomplished using FileMaker function calls from within FileMaker calculations. These calculations are generally determined from within FileMaker "SetField" or "If" script steps. This document describes the plug-in functions.

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) Reverse engineer our FileMaker demo file and review video tutorials

Demo and video tutorials can be found here: <http://www.productivecomputing.com/plugins/change-printer-id-79/>

- 4) Familiarize yourself with printing in FileMaker

II. Integration Steps

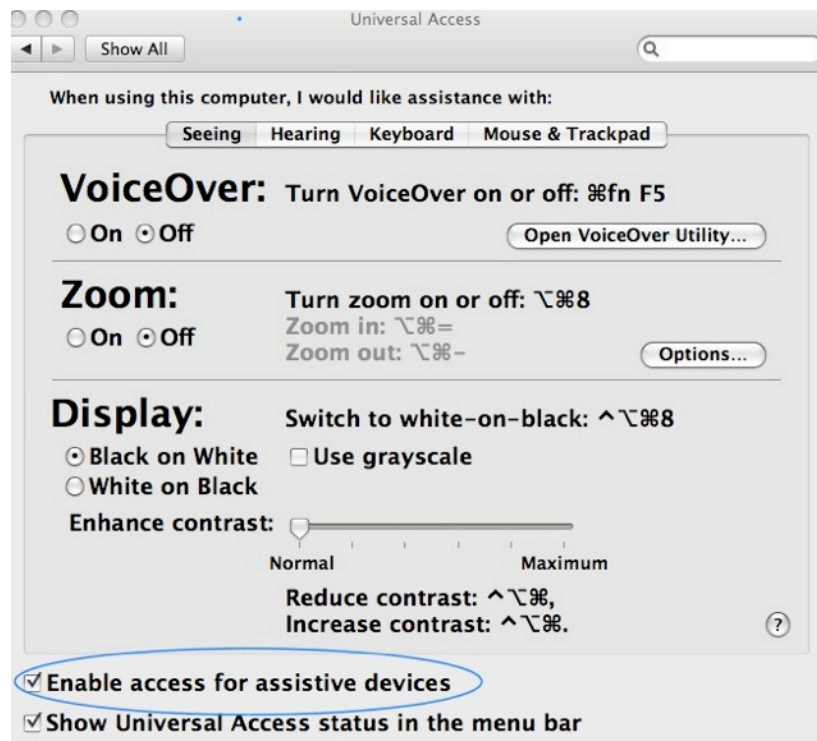
Accessing and using the plug-in involve the following steps.

1) Installing and Registering 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 Windows the plug-in will have a ".fmx" extension. 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 Windows this is normally located here: C:\Program Files\FileMaker\FileMaker X\Extensions. 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
- 5) On a Mac, then please ensure that access for assistive devices is enabled as shown below. This can be found by navigating to "System Preferences" then "Universal Access".



Registering the Plug-in:

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

- 6) Confirm that you have access to the internet and open our FileMaker demo file, which can be found in the "FileMaker Demo File" folder in your original download.
- 7) 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 our FileMaker Demo file on the Setup tab.
- 8) 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 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-in 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 Windows this certificate is in the form of a ".pci" file. On Mac this certificate is in the form of a ".plist" file.

How do I hard code the registration process?

You can hard code the registration process inside a simple "Plug-in Checker" script. The "Plug-in Checker" script should be called at the beginning of any script using a plug-in function and uses the PCCC_Register, PCCC_GetOperatingMode and PCCC_Version functions. This eliminates the need to manually register each machine and ensures that the plug-in is installed and properly registered. Below are the basic steps to create a "Plug-in Checker" script.

```
If [ PCCC_Version( "short" ) = "" or PCCC_Version( "short" ) = "?" ]
Show Custom Dialog [ Title: "Warning"; Message: "Plug-in not installed."; Buttons: "OK" ]
If [ PCCC_GetOperatingMode ≠ "LIVE" ]
Set Field [Main::gRegResult; PCCC_Register( "licensing.productivecomputing.com" ; "80" ; "\PCIReg/pcireg.asp" ;
"your license ID" )
If [ Main::gRegResult ≠ 0 ]
Show Custom Dialog [ Title: "Registration Error"; Message: "Plug-in Registration Failed"; Buttons: "OK" ]
```

2) Changing Printers

The PCCP_ChangePrinter(PrinterName ; optShowDlg ; optPause ; optCopies ; optSource ; optOrientation) function is used to change the name of the printer in FileMaker to which the print job is sent and set optional attributes. The function has six parameters. The first parameter "PrinterName" is the name of desired printer. The second parameter "optShowDlg" tells the plug-in that it will change the printer, but will allow the end user to initiate the print job. The third parameter "optPause" allows the print dialog to be displayed for the desired number of milliseconds. The fourth parameter "optCopies" determines the number of copies the job will print. The fifth parameter "optSource" determines which source tray the paper will come from on the printer. The sixth parameter "optOrientation" determines the orientation of the page that prints and is only applicable on a Windows machine as on a Mac you can use the FileMaker "PrintSetup" script step to select the orientation. All parameters are explained in further details in the "Functions Guide."

The PCCC_ChangePrinter function is used in conjunction with FileMaker's Print[] script step. Immediately after calling the PCCC_ChangePrinter function a subsequent call to FileMaker's Print[] script step should be called.

Before calling the PCCC_ChangePrinter function it is recommended that you obtain a list of all printers available. The PCCP_GetPrinterAt(index) function returns the name of a printer located in the systems internal list of printers. The value returned by the function is valid to use with the PCCP_ChangePrinter function.

The easiest way to describe how to use these functions to change printers and get valid printer names is to show some example scripts for different scenarios.

In the example scripts below these assumptions are made:

- A layout named 'Printable Layout' exists in the FileMaker solution
- There is a table named 'Main' with a global variable named 'gResult'
- There is a table named 'Printers' with a field named 'Names'
- One of the names returned by PCCP_GetPrinterAt is "\\Brother on DC1"

3) Get Valid Printer Names

The first step with any scenario is to decide to which printer you wish to print. The valid names of available printers can be retrieved with the PCCP_GetPrinterAt(index) function. This function is used to determine the exact name of a printer as it will be used in the PCCP_ChangePrinter function. To demonstrate using the function the following example script iterates through all of the printers on the local machine and grabs the appropriate name. Each name is stored in its own record in a "Printers" table.

```
###  
Go To Layout[ Printers ]  
Show All Records  
Delete All Records[ No Dialog ]  
Set Field[ Printers::Counter ; 1 ]  
Loop  
  New Record  
  Set Field[ Printers::Name ; PCCP_GetPrinterAt( Printers::Counter ) ]  
  Exit Loop If [ Left( Name ; 9 ) = "!!ERROR!!" ] //there are no more printers  
  Set Field[ Printers::Counter ; Printers::Counter + 1 ]  
End Loop  
#remove the last record as it does not hold a printer name  
Delete Record[ No Dialog ]  
###
```

Each record in the Printers table now holds a valid name for a printer. Any of these names can be used in the PCCP_ChangePrinter function call.

4) Automatic Printing

You have the option of printing automatically without requiring user interaction to press "OK" in the print dialog screen. This is accomplished by omitting the second parameter in the ChangePrinter function or by passing "False" or 0 as the second parameter. This can be written one of the following four ways:

- a. PCCP_ChangePrinter("\\Brother on DC1")
- b. PCCP_ChangePrinter("\\Brother on DC1" ; "")
- c. PCCP_ChangePrinter("\\Brother on DC1" ; "False")
- d. PCCP_ChangePrinter("\\Brother on DC1" ; "0").

Please note that although the function "presses" the print button before quitting, the user will briefly see the print dialog displayed by FileMaker.

In the following example script below the first step navigates to the proper layout. The next step then sets the desired printer. Then we finally print the document. The plug-in will dismiss the print dialog as soon as the printer has been properly changed and automatically print the document on the desired printer without user interaction.

```
###  
Got To Layout[ "Printable Layout" (Main) ]  
Set Field[ Main::gResult ; PCCP_ChangePrinter( "\\Brother on DC1" ) ]  
Print[]  
Go To Layout[ original layout ]  
###
```

5) Automatic Printer Selection Without Printing

You also have the option of selecting the printer without printing. This will require user interaction to press "OK" in the print dialog screen. This is accomplished by passing "True" or 1 as the second parameter in the ChangePrinter function. This is written the following way: PCCP_ChangePrinter("\\Brother on DC1" ; "True") or PCCP_ChangePrinter("\\Brother on DC1" ; "1"). The function quits after setting the printer name.

This proves desirable when your want to select the printer for the end user, but want the end user to actually select "OK" to send the print job to the printer. In the following example script we add a second parameter to the PCCP_ChangePrinter function which tells the plug-in to only select the proper printer but does not dismiss the dialog. This lets the end user make any other adjustments to the job before sending it to the printer.

```
###
Got To Layout[ "Printable Layout" (Main) ]
#pass a boolean 'true' value (True or 1) in the second parameter
# This leaves the Print dialog on the monitor
SetField[ Main::gResult ; PCCP_ChangePrinter( "\\Brother on DC1" ; 1 ) ]
Print[]
Go To Layout[ original layout ]
###
```

6) Optional Printing Parameters or Attributes

You also have the option of setting optional parameters giving you more flexibility over the various printing attributes. The PCCP_ChangePrinter(PrinterName ; optShowDlg ; optPause ; optCopies ; optSource ; optOrientation) function now offers parameters to add a pause into the print job, select the number of copies to print, select the source paper tray and determine the orientation of the page. Let's briefly explore these parameters and for a complete detailed explanation see the "Functions Guide."

optPause sets the desired number of milliseconds to pause the print dialog before the print job proceeds.

optCopies determines the number of copies the job will print.

optSource determines which source tray the paper will come from on the printer.

optOrientation determines the orientation of the page that prints and is only applicable on a Windows machine as on a Mac you can use the FileMaker "PrintSetup" script step to select the orientation.

For example:

```
PCCP_ChangePrinter( "\\Brother on DC1" ; 0 ; 1000 ; 3 ; "Tray 1" ; 0 ) or
PCCP_ChangePrinter( "\\Brother on DC1" ; 1 ; "" ; 5 ; "Tray 2" ; 1 )
```

7) Get or Set System Printer

You also have the option of getting or selecting the default system printer. This is accomplished using either the PCCP_GetSystemPrinter or PCCP_SetSystemPrinter functions.

We recommend reading the "Functions Guide" in order to realize the full potential of the plug-in functions. If there is missing functionality that you would like to have in the plug-in, please feel free to e-mail a request to support@productivecomputing.com.

III. Error Handling

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 PCCC_GetLastError (Type) function for a detailed description of what the exact error was.

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: " & PCCC_GetLastError ]
```

```
End If
```

The PCCC_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 PCCC_GetLastError ("Number"), then an error number will be returned. If format parameter is empty such as PCCC_GetLastError or PCCC_GetLastError(""), then an English error description will be returned.

Please find a list of return codes and descriptions below for your reference.

Error Number	Error Text
-1	Plug-in not registered or session expired
-3	Invalid # of Parameters
-4	Invalid Parameter value(s)
-10	Failed Registration
-5000	Specified printer not found
-5001	Index out of bounds
-5002	Unable to get system printer
-5003	Still waiting for last call to finish
-5004	No printers listed
-5005	Unable to set system printer

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!