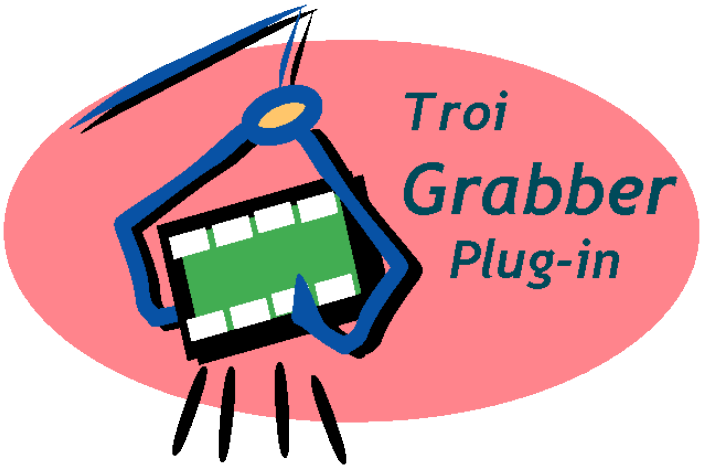


Troi Grabber Plug-in 2.3

USER GUIDE

April 2012

Troi Automatisering

Boliviastraat 11

2408 MX Alphen a/d Rijn

The Netherlands

You can also visit the Troi web site at: <http://www.troi.com/> for additional information.

Table of Contents

Installing plug-ins

Starting with FileMaker Pro 12 a plug-in can be installed directly from a container field. Please see the **EasyInstallTroiPlugins.fmp12** example file to install plug-ins with FileMaker Pro 12.

The instructions below show FileMaker 10. You can also install the plug-in with FileMaker Pro 9 and 11.

For Mac OS X:

- Quit FileMaker® Pro.
- Put the file “Troi_Grabber.fmplugin” from the folder “Mac OS Plug-in” into the “Extensions” folder in the FileMaker Pro application folder.
- If you have installed previous versions of this plug-in, you are asked: “An older item named “Troi_Grabber.fmplugin” already exists in this location. Do you want to replace it with the one you’re moving?’. Press the OK button.
- Start FileMaker Pro. The first time Troi Grabber Plug-in is used it will display a dialog box, indicating that it has loaded and showing the registration status.

For Windows:

- Quit FileMaker Pro.
- Put the file "Troi_Grabber.fmx" from the directory "Windows Plug-in" into the "Extensions" subdirectory in the FileMaker Pro application directory..
- If you have installed previous versions of this plug-in, you are asked: “This folder already contains a file called 'Troi_Grabber.fmx'. Would you like to replace the existing file with this one?’. Press the Yes button.
- Start FileMaker Pro. The Troi Grabber Plug-in will display a dialog box, indicating that it is loading and showing the registration status.

TIP You can check which plug-ins you have loaded by going to the plug-in preferences: Choose **Preferences** from the **Edit** menu, and then choose **Plug-ins**.

You can now open the file "All Grabber Examples.fp7" to see how to use the plug-in's functions. There is also a function overview in the download.

If you have problems

This user guide tries to give you all the information necessary to use this plug-in. So if you have a problem please read this user guide first. You may also visit our support web page:

<http://www.troi.com/support/>

This page contains FAQ's (Frequently Asked Questions), help on registration and much more. If that doesn't help you can get free support by email. Send your questions to **support@troi.com** with a full explanation of the problem. Also give as much relevant information (version of the plug-in, which platform, version of the operating system, version of FileMaker Pro) as possible.

If you find any mistakes in this manual or have a suggestion please let us know. We appreciate your feedback!

TIP You can get more information on returned error codes from the OSErrrs database on our web site: <http://www.troi.com/software/oserrrs.html>. This free FileMaker database lists all error codes for Windows and Mac OS!

What can this plug-in do?

The Troi Grabber Plug-in adds video image grabbing functions. With it you can take a picture from a video camera and put it into a container field. You can also record a movie, get timecode info and more.

Software requirements

System requirements for Mac OS X

Mac OS X 10.5.x (Leopard), Mac OS X 10.6.x (Snow Leopard), Mac OS X 10.7 (Lion)
QuickTime 7 or later
You also need:

- a QuickTime compatible video device
- a QuickTime driver for this video device

Mac OS X comes with drivers for quite a few devices, like DV camcorders via firewire. Also webcams like the built-in iSight should work.

System requirements for Windows

Windows XP (Service Pack 2) on Intel-compatible computer, Pentium III 500MHz or higher
Windows Vista on Intel-compatible computer, Pentium III 800MHz or higher
Windows 7 on Intel-compatible computer 1 GHz or faster.

You also need:

- a TWAIN compatible input source, like a video camera (CamCorder) or a scanner
- installed TWAIN drivers for this input source

FileMaker Pro requirements

FileMaker Pro 10 or FileMaker Pro Advanced 10 or higher.
FileMaker Pro 11 or FileMaker Pro Advanced 11 or higher.
FileMaker Pro 12 or FileMaker Pro Advanced 12 or higher.

NOTE Troi Grabber Plug-in will probably run fine with FileMaker 7, 8.x and 9, but we have not tested this and we don't provide support for this. Also note that Troi Grabber Plug-in, with older FileMaker versions, will probably run on older operating systems for example Mac OS X 10.3.9 and Windows 2000 however we also do not support this.

NOTE 2 Troi Grabber Plug-in version 2.0 (and later) started using the native plug-in syntax introduced with FileMaker Pro 7. This means that the functions of this plug-in have this format: FunctionName(parameter1 ; parameter2). This native plug-in syntax makes it possible to support Unicode and more.

Troi Grabber Plug-in version 2.x does **not** run on versions prior to FileMaker Pro 7.0. If you need to run on versions prior to FileMaker Pro 7: see our web site for the Troi Grabber Plug-in 1.5 which is using the 'classic' plug-in API, which is using the External("functionName" , "parameter") format.. See our web site here:
<http://www.troi.com/software/grabberplugin.html>

FileMaker Server requirements and AutoUpdate

FileMaker Server 10 or FileMaker Server Advanced 10 or higher.
FileMaker Server 11 or FileMaker Server Advanced 11 or higher.

You can use FileMaker Server to serve databases that incorporate functions of the Troi Grabber Plug-in (**client-side**): You need to

have the plug-in installed at the clients that use these functions.

The **AutoUpdate** feature of FileMaker Server 10 or 11 can help you automate installing and updating plug-ins automatically. We created an example file and a tar formatted plug-in of Troi Grabber Plug-in (only needed on Mac OS X) to get you started. Visit our AutoUpdate web page to download the example:

<http://www.troi.com/software/autoupdate.html>

NOTE Troi Grabber Plug-in can **NOT** be used by FileMaker Server as a server-side plug-in or as a plug-in used by the web publishing engine, as the plug-in requires Graphic User Interface (GUI), which is not possible with FileMaker Server. For more info see our web site here:

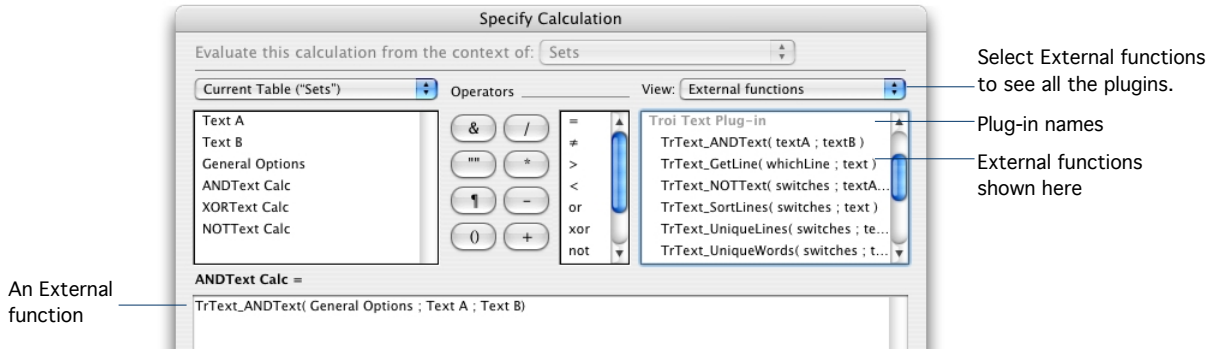
<http://www.troi.com/support/filemaker-server-side-plug-ins.html>

NOTE 2 With FileMaker Pro 12 the **AutoUpdate** feature of FileMaker Server has been removed, as plug-ins can be installed directly from a container field. See the **EasyInstallTroiPlugins.fmp12** example file to install plug-ins with FileMaker Pro 12.

Getting started

Using external functions

The Troi Grabber Plug-in adds new functions to the standard functions that are available in FileMaker Pro. The functions added by a plug-in are called external functions. You can see those extra functions for all
plug-ins at the top right of the Specify Calculation dialog box:



You use special syntax with external functions: FunctionName(parameter1 ; parameter 2) where FunctionName is the name of an external function. A function can have zero or more parameters. Each parameter is separated by a semi-colon. Plug-ins don't work directly after installation. To access a plug-in function, you need to add the calls to the function in a calculation, for example in a calculation in a ScriptMaker Script.

Where to add the external functions?

External functions for this plug-in are intended to be used in a Set Field script step using a calculation. For most functions of this plug-in, it makes no sense to add them to a define field calculation, as the functions will have side effects.

Simple example

We start with a simple example to get you started. Create a new database, with a container field called `myImageField`. Create a new ScriptMaker Script called "Grab My Image". Delete all steps and then add the following script step:

```
Set Field [ myImageField, Grab_OpenImageGrabWindow( "-Unused" ; "QTKit"; "iSight" )]
```

This function will show a dialog where the user can grab an image. The result is the image, which can be placed in a container field.

NOTE Function names, like `Grabber_OpenImageGrabWindow`, are no longer case sensitive. You can type them or get them from the External Functions list at the top right of the "Specify Calculation" dialog.

Please take a close look at the included example files, as they provide a great starting point. From there you can move on, using the functions of the plug-in as building blocks. Together they give you great new tools to grab images in your solution!

Summary of functions

The Troi Grabber Plug-in adds the following functions:

function name	short description
Grabber_Version	Checks for correct version of the plug-in. This function is also used to register the plug-in.
Grabber_VersionAutoUpdate	Returns standard version number for AutoUpdate of FileMaker Server.
Grab_GetDeviceInfo	Returns information about available technology and devices
Grab_OpenImageGrabWindow	Shows a window for grabbing a single still image.
Grab_OpenMovieGrabWindow	Shows a window for grabbing a movie.
Grab_SetWindowTitle	Sets the name of the grabber dialog window
Grab_SetWindowPosition	Sets the position on the screen of the grabber dialog window to be shown.
Grab_SetWindowSize	Sets the size of the grabber dialog window to be shown.
Grab_Initialise	Test if there is a video input source available and initialise it.
Grab_StartPreview	Starts the video preview stream at the specified coordinates and dimensions.
Grab_GrabImage	This grabs the current image from the video preview as a container.
Grab_RecordMovie	This will record the movie and write it to the specified file.
Grab_DoSettingsDialog	This shows the video settings dialog. With this dialog the user can change the settings of the video source.
Grab_GetCurrentSettings	This returns the current video settings (as binary).
Grab_SetSettings	This sets the video settings of the Grabber to the given videoSettings.
Grab_SetRotation	This tells the grabber how to rotate the video preview.
Grab_DisplayCropRect	Show a cropping rectangle on the video source.
Grab_GetTimecode	Return a timecode of a movie.
Grab_CropImage	Crops an image to smaller dimensions.
Grab_StopPreview	Stops the video preview stream.

About technologies

Most functions of Troi Grabber Plug-in now have a technology parameter. This parameter makes it possible to extend functionality and add support for different technologies to the plug-in in the future.

Currently Troi Grabber Plug-in supports these 3 technologies:

- QTKit — A modern QuickTime implementation, using a separate window.
- QuickTimeClassic — Classic QuickTime, as used by our Grabber for FileMaker 6 and earlier. Supports Video Preview inside of the FileMaker window.
- TWAIN — on Windows only

NOTE: although both QuickTimeClassic and QTKit are based on QuickTime API, they support different cameras and image formats. For example: with the same camera, QuickTimeClassic only supports the DV format and not HDV format while QTKit does support this.

NOTE HDV support is only available with QTKit technology and Final Cut Studio installed.

Implementing image grabbing

Implementing video grabbing it not difficult, but the developer of the database must be aware of some limitations to get optimum results.

Plug-in limitations

The Grabber plug-in has some limitations. Please be aware of the following:

Limitations for Mac OS X and QuickTimeClassic

- **Preview only when window is on top**: The video preview is only shown and updated when FileMaker is the front most application. Also the window where the preview is shown must be front most. If you switch to a different file or application the video preview is no longer updated.

- **FileMaker is unaware of the video preview**: When you start a video preview, the plug-in shows the image stream in the window. The FileMaker application is not aware that this is happening. There is no fixed relation to the preview rectangle and the other FileMaker objects on the screen. So when a user hides the status bar the FileMaker objects move to the left but the preview rectangle is still shown on the same coordinates. This is not serious, but looks very strange and may confuse a user. Therefore before starting a video preview, be sure to lock the window down. Use these script steps:

```
Toggle Status Area [Hide, Lock]
Set Zoom Level [100%, Lock]
```

It doesn't matter if the status area is hidden or shown, but make sure to lock it. The same holds true for the Zoom level; choose a fixed zoom level and lock it.

- **Video Preview is not aware of the layout**: Make sure to stop the video preview when the user leaves the layout. The preview rectangle is still shown on the same coordinates if you don't do this. This is also not serious, but looks very strange and may confuse a user. Also note that when switching to Layout mode the preview can continue to be shown. At the moment we don't have a way to prevent this surprising but harmless behaviour.

- **Video Preview must fit on the window**: Make sure the video preview is completely visible on the screen. Otherwise the capture is only of the visible part of the rectangle.

Steps for creating an image grabbing database (QuickTimeClassic)

These are the main steps to create a grabbing database:

1 - create a picture container field and some assisting global fields.
2 - create a new layout with room for the video input rectangle and the picture field on it. You can also modify an existing layout.
3 - create 3 scripts: to start the preview, to capture an image and to stop the preview.
4 - if wanted you can do this in a loop so you grab pictures in sequence.

1- Define Fields

In your database define the following fields:

```
image           Container field
gErrorCode      Global, Text
gWidthHeight    Global, Text
gWidth          Global, Text
gHeight         Global, Text
```

2- Create a Grabber layout

Create a layout or modify an existing layout. Make room for the video preview image. You can do this by trying out the preview and adjusting the scripts and layout. Create a rectangle and name this rectangle object on the layout. Call it "previewrect" in the object info dialog box. To show this dialog (in layout mode): first select the rectangle on the layout, then View->Object Info.

3- Create two Grabber scripts

In ScriptMaker define a script "Start Preview (Mac)". This script will get the width and height from the video source and starts the video preview:

Define the script "Start Preview (Mac)" as follows:

```
#Start up the preview of the image to be grabbed
#First: initialise and get the dimensions of the attached camera:
Set Field [ GrabImage::gWidthHeight; Grab_Initialise( "-Unused" ; "QuickTimeClassic" ) ]
If [ Left(GrabImage::gWidthHeight ; 2)  <>  "$$" ]
    #Find the dimensions of the named rectangle object:
    Set Variable[$previewrect_left;
                        GetLayoutObjectAttribute("previewrect"; "left") - Get ( WindowLeft )]
    If [ $previewrect_left <> "" ]
        #we found the preview rectangle on the current layout.
        Set Variable[ $previewrect_top;  Get( WindowTop ) ]
        Set Variable[ $previewrect_top;
                GetLayoutObjectAttribute("previewrect" ; "top" ) - Get( WindowTop ) - 20 ]
        Set Variable[ $previewrect_right;
                GetLayoutObjectAttribute("previewrect" ; "right" )  - Get ( WindowLeft ) ]
        Set Variable [ $previewrect_bottom;
                GetLayoutObjectAttribute("previewrect" ; "bottom" )  - Get ( WindowTop ) - 20 ]
        Set Variable [ $previewrect_bounds;
                            $previewrect_left & " " &
                            $previewrect_top & " " &
                            $previewrect_right & " " &
                            $previewrect_bottom ]
        Set Field[ GrabImage::gErrorCode;
           Grab_StartPreview( "-MaintainProportions" ; "QuicktimeClassic" ; $previewrect_bounds  )]
    Else
        Show Custom Dialog[ "Could not get LayoutObjectAttribute of the PreviewRect."; Buttons: "OK"]
    End If
Else
```

```
    #An Error occurred. put the errorCode into the global.
    Set Field [ GrabImage::gErrorCode; GrabImage::gWidthHeight ]
    Perform Script [ "Error Message Handler" ]
End If
```

In ScriptMaker also define a script "New record and Grab Image" as follows:

```
Set Variable[ $theImage; Value:Grab_GrabImage(" -Unused " ; "QuickTimeClassic") ]
Set Field [ GrabImage::gErrorCode; GetAsText($theImage) ]
If [ Left(GrabImage::gErrorCode ; 2 )  <> "$$" ]
     Set Field [ GrabImage::gErrorCode; 0 ]
     New Record/Request
     Set Field [ GrabImage::Image; $theImage ]
Else
     Perform Script [ "Error Message Handler" ]
     Halt Script
End If
Go to Field [  ]
```

This script will capture the current preview image in a container in a new record. You can of course change this to paste it in an existing container.

4- Grab Pictures in a loop

If you want to grab images in a loop define a script "Loop: Grab Images into records" as follows:

```
Go to Layout [Image grabbing]
Loop
     Perform Script [Sub-scripts, New record and Grab Image]
     Pause/Resume Script [0:00:10]
     Exit Loop If [gErrorCode <> 0]
End Loop
```

This will grab an image every 10 seconds.

Rotation and cropping

The following features of Troi Grabber plug-in are only available for Mac OS X, as they make use of QuickTime. In the picture to the right a sample preview is showing a video preview that is rotated 90° and it shows a cropping rectangle.

Implementing rotation and cropping

The Troi Grabber Plug-in for Mac OS can rotate the video preview and crop images. Implementing these features require that you add some special steps to your ScriptMaker scripts.

These are the main steps to take to enable these features:

1 - Initializing the video preview using a "GWorld"
2 - Set the rotation
3 - Initialize the cropping rectangle

NOTE You don't need to implement both step 2 and 3. You can also use rotation or cropping separately.

What is a GWorld?

The cropping and rotation functions are implemented with the help of a GWorld. A GWorld is a technical term: it is an offscreen part of the computer memory that makes it possible to manipulate the grabbing process.

Initializing the video preview

You don't need to remember what a GWorld is. The main thing to remember is that for Rotation and Cropping you need to initialise the Grabber Plug-in using the `-UseGWorld` switch, like this:

```
Set Field [gWidthHeight, Grab_Initialise( "-UseGWorld"; "QuickTimeClassic" ))]
```

TIP A GWorld uses extra memory, so for large video images you might need to give FileMaker Pro more memory.

Setting the rotation

If you want to rotate the video image generated you use the function `Grab_SetRotation`. Here is a sample script `Set rotation 90`:

```
# Set rotation to 90 degrees:
Set Field [gErrorCode, Grab_SetRotation( "-Unused" ; "QuickTimeClassic"; "90")]
```

Initializing the cropping rectangle

If you want to show a cropping rectangle use the function `Grab_DisplayCropRect`. Here is a sample script `Show Crop Rectangle`:

```
# Show a rectangle at the specified coordinates:
Set Field [gErrorCode, Grab_DisplayCropRect("-Unused" ;
                                    "QuickTimeClassic"", "100 50 340 370")]
```

The parameters for the `Grab_DisplayCropRect` function are the 4 coordinates of the cropping rectangle. The bounds of the cropping rectangle are relative to the video preview. They are in this order: "left top right bottom", which is the same order as the FileMaker FieldBounds function

Example script

The following script `Start Preview Mac RotCrop` shows the elements together:

```
# Start up the preview of the image to be grabbed:
Set Field [gWidthHeight, Grab_Initialise( "-useGWorld" ; "QuickTimeClassic" )]
If [Left(gWidthHeight , 2) <> "$$"]
      Set Field [gWidth, Left(gWidthHeight , Position(gWidthHeight , "|" , 1 , 1 ) - 1)]
      Set Field [gHeight, Right(gWidthHeight , Position(gWidthHeight , "|" , 1 , 1 ) + 1)]
      Set Field [gErrorCode, Grab_StartPreview("-Unused" ; "QuickTimeClassic" ;
                  "10 100 " & GetAsText(10 + gHeight / 2) & " " & GetAsText(100 + gWidth /2)]
      Perform Script [Sub-scripts, Set Grabber Settings]
      Perform Script [Sub-scripts, Set rotation 90]
      Perform Script [Sub-scripts, Show Crop Rectangle]
Else
      # An error occurred. put the error code into the global:
      Set Field [gErrorCode, gWidthHeight]
End If
```

After this script has run the preview should be visible, rotated 90° and with a cropping rectangle.

Getting a cropped image into a container

When you are previewing you can grab the complete image, as well as a part of the image into a container field.
There are 2 methods to crop images:

1 - Crop an image from the container
2 - Get a cropped image from the video preview

Use the first method if it is important that the cropped image is exactly the same as (the part of) the larger image. If you use the second way the cropped image is grabbed freshly from the video source and will be the image that is then showing.

Scenario 1

Storing 2 identical images: 1 full size and 1 cropped

If you want the grabber to grab an image and also store a part of that same image, use this script.

"New record + store 2 identical images: 1 Full + 1 Cropped":

```
## 1- get the full size image from the grabber into the container...
Set Variable [ $theImage; Value:Grab_GrabImage(" -Unused " ; "QuickTimeClassic") ]
Set Field [ GrabImage::gErrorCode; GetAsText($theImage) ]
If [ Left(GrabImage::gErrorCode ; 2 )  <> "$$" ]
  Set Field [ GrabImage::gErrorCode; 0 ]
  New Record/Request
  Set Field [ GrabImage::Image; $theImage ]
  ## 2- crop the image in the container....
  Perform Script [ “Crop Image in container” ]

Else
  Perform Script [ “  Error Message Handler” ]
  Halt Script
End If
Go to Field [  ]
```

This is the script `Crop Image in container`:

```
# Crop an image in a container to the requested size.
If [ $$cropRectangle <> "" ]
  Set Field [ GrabImage::Image Detail;
               Grab_CropImage( "-Unused" ; GrabImage::Image ; $$cropRectangle ) ]
Else
  #Fall back on a default.
  Set Field [ GrabImage::Image Detail;
                     Grab_CropImage( "-Unused" ; GrabImage::Image ; "100 50 340 370" ) ]
End If
Set Field [ GrabImage::gErrorCode; GetAsText(GrabImage::Image Detail) ]
If [ Left(GrabImage::gErrorCode ; 2) <> "$$" ]
  Set Field [ GrabImage::gErrorCode; 0 ]
Else
  Perform Script [ “  Error Message Handler” ]
  Halt Script
End If
```

TIP In this example the parameters come as a global variable into the script.

Scenario 2

Getting full size image and cropped image after each other

If you want the grabber to grab a full size image first then get a cropped image after that use this script.

"New record and Grab 2 Images To Container":

```
## 1- get the full size image from the Grabber into the container...
Set Variable [ $theImage; Value:Grab_GrabImage(" -Unused " ; "QuickTimeClassic") ]
Set Field [ GrabImage::gErrorCode; GetAsText($theImage) ]
If [ Left(GrabImage::gErrorCode ; 2 )  <> "$$" ]
  Set Field [ GrabImage::gErrorCode; 0 ]
  New Record/Request
  Set Field [ GrabImage::Image; $theImage ]
Else
  Perform Script [ “  Error Message Handler” ]
  Halt Script
End If
Go to Field [  ]
## 2- get a cropped image into the 2nd container field:
Set Variable [ $theImage; Value:Grab_GrabImage(" -Cropped " ; "QuickTimeClassic") ]
Set Field [ GrabImage::gErrorCode; GetAsText($theImage) ]
If [ Left(GrabImage::gErrorCode ; 2 )  <> "$$" ]
  Set Field [ GrabImage::gErrorCode; 0 ]
  Set Field [ GrabImage::Image Detail; $theImage ]
  Go to Field [  ]
Else
  Perform Script [ “  Error Message Handler” ]
  Halt Script
End If
```

Recording movies (QTKit)

The easiest way to record a movie from FileMaker script is with the Grab_OpenMovieGrabWindow function. Just add this step to your script:

```
Set Field[ gErrorCode, Grab_OpenMovieGrabWindow( "-Unused" ; "QTKit"; "deviceName" ;
                            "MacHD:MyFolder:testmovie.mov" )]
```

See the example file RecordMovie_QTKit.fp7 for more details on selecting a video source and selecting an movie file.

Recording movies (QuickTimeClassic)

With the QuickTimeClassic technology the function `Grab_RecordMovie` can also record a video stream to a movie file. This is only available on Mac OS X, as this uses QuickTime.

- Recording only works if no GWorld is used. So make sure you don't use the switch `useGWorld` when you initialise the grabber with the function `Grab-Initialise.`

- The Troi Grabber plug-in can not select or manipulate files. Our Troi File plug-in makes it easy to work with movie files on disk. You can download a demo copy from our web site on this page:

<http://www.troi.com/software/fileplugin.html>

These are the main steps to create a database that can record movies:

1 - create a container field and some assisting global fields.
2 - create a new layout with room for the video input rectangle and the container field on it. You can also modify an existing layout.
3 - create 3 scripts: to start the preview, to record a movie and to stop the preview.

Steps 1 to 3 are the same as the steps for creating a picture grabbing database with the exception of the script that records a movie. See "Steps for creating a picture grabbing database (Mac OS)" on page 6 for all these steps.

NOTE if you implement all steps of "Steps for creating a picture grabbing database (Mac OS)" and the script below you can make a database that can grab single images and also record a movie too.

Script variant 1: Record Movie until mouse click

This script will record until the user clicks on the mouse. In addition to the existing field in your database define a global text field `gInfoText`. This field will indicate to the user what to do. In ScriptMaker also define a script "Record Movie until mouse click" as follows:

```
#This will create movie file and record into it until the user clicks on the mouse.
Enter Browse Mode []
Set Field [gInfoText, "Recording movie. Click the mouse to stop recording."]
Refresh Window []
#The next step is to make sure that on Mac OS X the screen is updated correctly:
Pause/Resume Script [0:00:00]
Set Field [gErrorCode,  Grab_RecordMovie( "-RecordUntilMouseClick " &
                              gOverwriteExistingFiles ; "QuickTimeClassic"; gPathToMovieFile)]
Set Field [gInfoText, ""]
If [gErrorCode = 0]
     Show Message [The movie has been recorded and written to disk.]
Else
     Beep
     Show Message [An error occurred.]
End If
```

Script variant 2 : Record Movie during a specified time

This script will record for a specified number of seconds. In addition to the existing field in your database define a global text field `gInfoText`. This field will indicate to the user what to do. In ScriptMaker also define a script "Record Movie (specified time)" as follows:

```
#This will create movie file and record 10 seconds of movie.
Enter Browse Mode []
Set Field [gInfoText, "Recording 10 secs. of movie until: "  &
             TimeToText(Status(CurrentTime) + 10) &". Press ESC to stop earlier."]
Refresh Window []
Pause/Resume Script [0:00:01]
Set Field [gErrorCode,  Grab_RecordMovie( "-RecordSpecifiedTime " &
                    gOverwriteExistingFiles ; "QuickTimeClassic"; gPathToMovieFile) ; 10]
Set Field [gInfoText, ""]
If [gErrorCode = 0]
     Show Message [The movie has been recorded and written to disk.]
Else
     Beep
     Show Message [An error occurred.]
End If
```

Function Reference

Grab_CropImage

Syntax Grab_CropImage(switches; image; cropBounds)

Crops an image to smaller dimensions.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
image	the image to be cropped
cropBounds	the cropping rectangle relative to the image. Dimensions (in pixels) are in this order: "left top right bottom", which is the same order as the FileMaker FieldBounds function

Returned result

If successful it returns the cropped imaged. If unsuccessful it returns an error code starting with $$ and the error code.

Special considerations

This function is currently only available on Mac OS X.

The original image name is now preserved. If the image has no name it is set to "GrabberCroppedImage.jpg". The image name will be used by the FileMaker application for an initial suggested filename when exporting field contents.

Example usage

Grab_CropImage("-Unused"; imageField; "100 50 240 320")

This will crop the image to a rectangle of the specified coordinates.

Grab_DisplayCropRect

Syntax Grab_DisplayCropRect(switches; technologyName; cropBounds)

This tells the grabber to show a cropping rectangle on the video source.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
technologyName	the technology to use
cropBounds	the bounds of the cropping rectangle to show, relative to the video preview. Dimensions (in pixels) are in this order: "left top right bottom", which is the same order as the FileMaker FieldBounds function

Returned result

If successful it returns 0. If unsuccessful it returns an error code starting with $$ and the error code.

Special considerations

This can be used with the technology: QuickTimeClassic.
To stop the cropping rectangle use a blank cropBounds.
The coordinates of cropBounds are in the same order as the FileMaker FieldBounds function.
This function is currently only available on Mac OS X.

Example usage

Grab_DisplayCropRect("-Unused"; "QuickTimeClassic" ; "100 50 240 320")

This will show a rectangle on the specified coordinates.

EXAMPLE 2

Grab_DisplayCropRect("")

This will stop the display of the cropping rectangle.

Grab_DoSettingsDialog

Syntax Grab_DoSettingsDialog(switches ; technologyName)

This shows the video settings dialog. With this dialog the user can change the settings of the video source.

Parameters

switches	set to "-video". No other switches are supported at this moment.
technologyName	the technology to use

Returned result

If successful it returns 0.
If unsuccessful it returns an error code starting with $$ and the error code.
$$-1 = user canceled

Special considerations

This can be used with the technology: QuickTimeClassic.
After the user clicks on the OK button the settings are changed. You can retrieve the current settings with the function Grab_GetSettings.

Example usage

Grab_DoSettingsDialog("-Video" ; "QuickTimeClassic")

This will show the Video Settings dialog.

Grab_GetCurrentSettings

Syntax Grab_GetCurrentSettings(switches ; technologyName)

This returns the current video settings (as binary)

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
technologyName	the technology to use

Returned result

If successful this function returns the current settings for the video input. If unsuccessful it returns an error code starting with $$ and the error code.

The setting is coded and can't be read. Store the result for later use in (for example) a global field.

Special considerations

This can be used with the technology: QuickTimeClassic.
Do not change the returned settings. Only store them in a field so you can reset them later.
Important: The video preview must be running first.

Example usage

Grab_GetCurrentSettings("-Unused" ; "QuickTimeClassic")

Grab_GetDeviceInfo

Syntax Grab_GetDeviceInfo(switches ; { technologyName })

This returns information about available technology and devices

Parameters

switches	determine the behaviour of the function
technologyName	(optional) the technology to use

Switches can be one of this:

-AvailableTechnologies	returns which technologies can be used, like for example QTKit
-GetVideoDeviceList	returns the names of the (video)devices available with a technology

-GetVideoCompressionOptionsList" returns a list of possible video compression options (QTKit technology only).

Returned result

If successful this function returns information about the possible devices.

Special considerations

At the moment the returned technologies can be QuickTimeClassic or QTKit on Mac OS X and TWAIN on Windows. New grabbing technologies might be supported in a future version of the plug-in.

Example usage

Set Field[gGrabberSettings, Grab_GetDeviceInfo("-AvailableTechnologies")]

This step will return for example:

QTKit
QuickTimeClassic

Example 2

Set Field[gGrabberSettings, Grab_GetDeviceInfo("-GetVideoDeviceList" ; "QTKit")]

Will return the names of the video devices, for example:

HV20
Built-in iSight

Grab_GetTimecode

Syntax Grab_GetTimecode(switches ; moviefilePath)

This will return a timecode of the movie.

Parameters

switches	these determine which timecode is returned
moviefilePath	the path to the movie file

switches must be one of the following:

-StartOfMovie	get the timecode of the beginning of the movie
-EndOfMovie	get the timecode of the end of the movie
-CurrentTime	get the timecode at the current point in the movie
-Duration	get the duration of the movie in timecode

Returned result

If successful it returns a timecode. If unsuccessful it returns an error code starting with $$ and the error code. Possible error codes are:

$$-2020	QuickTime not available
$$-2032	No timecode track found
$$-43	The file was not found
$$-37	The name of the file is not correct

Other errors may be returned.

Special considerations

This function is currently only available on Mac OS X.
If the movie does not contain a timecode track only the duration timecode can be returned.

Example usage

Grab_GetTimecode("-StartOfMovie" ; "Mac HD:projects:test.mov")

This will return the timecode of the start of the specified movie, for example "01:23:59:29"

Example 2

This example will put the timecodes into fields. We assume that in your FileMaker file the following fields are defined:

TimeCode Start	Text
TimeCode End	Text
TimeCode Duration	Text
gFilePath	Global, text

gFilePath should contain the path to the movie file, for example "Mac HD:movies:shrek.mov". In ScriptMaker add the following script steps:

```
Set Field[TimeCode Start, Grab_GetTimecode("-StartOfMovie" ; gFilePath ) ]
```

Grab_GetTimecode

```
If[ Timecode Start = "$$-2032"]
    # no time code found
     Exit Script
Else
   Set Field[TimeCode End, Grab_GetTimecode("-EndOfMovie" ; gFilePath )]
   Set Field[TimeCode Duration, Grab_GetTimecode("-Duration" ; gFilePath )]
End if
```

This will get the timecodes for the movie specified by gFilePath. If the movie does not have a timecode track an error code of "$$-2032" will be returned.

Grab_GrabImage

Syntax Grab_GrabImage(switches; technologyName {; imageName })

This returns current image from the video preview as a container.

Parameters

switches	determine the behaviour of the function
technologyName	the technology to use
imageName	(optional) the name of the image to be stored in the container, default is GrabberImage.jpg

switches can be emtpy or one of this:

-NoCrop	(default) the whole image is returned
-Cropped	the image is cropped to the dimensions of the shown copping rectangle

If the switches does not contain the -Cropped switch, the whole image is returned.

Returned result

If successful it returns the image. If unsuccessful it returns an error code starting with $$ and the error code.

Special considerations

This can be used with the technology: QuickTimeClassic.
The video preview must be started, see Grab_StartPreview for this.
The image will be of type JPEG, with medium quality (compression factor of 0.6).

Example usage

Grab_GrabImage(" -Unused " ; "QuickTimeClassic" ; "myGrabImage04.jpg")

This will return the grabbed image. The name of the image is stored in the container result. When you later export the image this name will be used by the FileMaker application.

Example 2

Set Field[theCroppedImage, Grab_GrabImage(" -Cropped ")]

This will return the image, cropped to the current cropping rectangle and put it in the container field theCroppedImage.

Grab_Initialise

Syntax Grab_Initialise(switches ; technologyName)

Use this function to test if there is a video input source available and initialise it.

Parameters

switches	determine the behaviour of the function
technologyName	the technology to use

You can add one or more of these switches:

-UseGWorld	use an offscreen graphic GWorld
-UseAltBounds	use alternative bounds (if you see a white band)

Returned result

If successful the result is the width and height of the video source.
If unsuccessful it returns an error code starting with $$ and the error code. Possible codes are:

$$-2020	QuickTime not available
$$-230	No video input device found

Special considerations

This can be used with the technology: QuickTimeClassic.
See the section "Rotation and cropping" in the user guide for more information when to use the -UseGWorld switch.

Example usage

Set Field [result , Grab_Initialise("-Unused" ; "QuickTimeClassic")]

This initialises the plug-in for QuickTimeClassic. This will return the width and height of the video source, or an error code. For example this might return "640 480".

Example 2

Grab_Initialise("-useGWorld -useAltBounds" ; "QuickTimeClassic")

Grab_OpenImageGrabWindow

Syntax Grab_OpenImageGrabWindow(switches ; technologyName ; deviceName {; imageName })

This shows a window for grabbing a single still image. The live video image preview is shown and you can click on the "Grab" button. The image is returned.

Parameters

switches	these determine how this function works
technologyName	the technology to use
deviceName	the name of the Video device to use
imageName	(optional) the name of the image to be stored in the container, default is GrabberImage.jpg

You can also add these switches:

-DontAdjustForNonSquarePixels	don't correct for video images that have non square pixels
-ImageQuality=x.x	sets the compression factor of the saved image, 0.0 is the minimum and 1.0 is the maximum quality

Returned result

If successful it returns the grabbed image. If unsuccessful it returns an error code starting with $$ and the error code. Possible error codes are:

$$-1 user canceled.
$$-4243 kErrNotImplemented, not available with this technology

other errors may be returned.

Special considerations

This can be used with the QTKit technology on Mac OS X and with the TWAIN technology on Windows.
The image will be of type JPEG. Default compression factor is 0.9 (high quality).

Example usage

Grab_OpenImageGrabWindow("-Unused" ; "QTKit"; "HV20" ; "myGrabImage03.jpg")

This script step will show a window for grabbing a single still image (on Mac OS X). The live video image preview is shown. After the user clicks on the Grab button the window is closed and the grabbed image is returned as the result. The name of the image is stored in the container result. When you later export the image this name will be used by the FileMaker application.

Example 2

Set Variable [$result, Grab_OpenImageGrabWindow("-ImageQuality=0.35" ; "TWAIN"; "EPSON Scan")]

This (Windows) example will open a TWAIN device and wait for the user to click on the grab button. The result is the grabbed image, a JPEG, and the quality will be fairly low, but small in size.

Grab_OpenMovieGrabWindow

Syntax Grab_OpenMovieGrabWindow(switches ; technologyName ; deviceName ; movieFilePath)

This shows a window for grabbing a movie. The live video image preview is shown and you can start and stop the recording.

Parameters

switches	these determine how this function works
technologyName	the technology to use
deviceName	the name of the Video device to use
movieFilePath	the path to where the movie file must be created

You can add one or more of these switches:

-OverwriteExisting	overwrite an existing file without generating an error
-DontAdjustForNonSquarePixels	don't correct for video images that have non square pixels

You can also add one of these switches:

-Compress240SizeH264Video	H.264 codec, medium quality
-CompressSD480SizeH264Video	(default) H.264 codec, higher quality
-CompressLosslessAppleIntermediateVideo	lossless recording, big sizes

Returned result

If successful it returns 0. If unsuccessful it returns an error code starting with $$ and the error code. Possible errorcodes are:

$$-1	user canceled.
$$-4243	kErrNotImplemented, not available with this technology

other errors may be returned.

Special considerations

This can be used with the technology QTKit.
This function is currently only available on Mac OS X.
Not all compression formats may be available.

Example usage

Grab_OpenMovieGrabWindow("-Unused" ; "QTKit"; "HV20" ; "MacHD:MyFolder:testmovie.mov")

Grab_RecordMovie

Syntax Grab_RecordMovie(switches ; technology; movieFilePath ; {seconds})

This will record the movie and write it to the specified file.

Parameters

switches	these determine how this function works
technologyName	the technology to use
movieFilePath	the path to the movie file to create
Seconds	(optional) the number of seconds of movie that will be recorded.

switches can be one of the following:

-RecordUntilMouseClick	(default) record until the user clicks the mouse
-RecordSpecifiedTime	record for the specified number of seconds

You can also add this switch:

-OverwriteExisting	overwrite an existing file without generating an error

Returned result

If successful it returns 0. If unsuccessful it returns an error code starting with $$ and the error code. Possible error codes are:

$$-2020	QuickTime not available
$$-2042	dataNotOpenForRead , start the preview first
$$-48	The file already exists
$$-4243	kErrNotImplemented, not available with this technology

Other errors may be returned.

Special considerations

This can be used with the technology: QuickTimeClassic.
This function is currently only available on Mac OS X.

Example usage

Set Field [result, Grab_RecordMovie("-RecordUntilMouseClick" ; "QuickTimeClassic"; "Mac HD:test.mov")]

This will record a movie "test.mov" on the harddisk. The movie will be recorded until a mouse is clicked.

Example 2

We assume that in your FileMaker file the following fields are defined:

gFilePath	Global, text
gRecordTime	Global, time

gFilePath should contain the path to the movie file, for example "Mac HD:movies:test.mov". gRecordTime should contain the number of seconds to record. In ScriptMaker add the following script step:

Set Field[gErrorCode, Grab_RecordMovie("-RecordSpecifiedTime" ; "QuickTimeClassic"; gFilePath; gRecordTime)]

Grab_RecordMovie

This will record a movie at the place specified by gFilePath. The movie will be recorded for the number of seconds in the gRecordTime field.

Grab_SetRotation

Syntax Grab_SetRotation(switches ; technology ; rotationAngle)

This tells the grabber how to rotate the video preview.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
technologyName	the technology to use
rotationAngle	the number of degrees the image must be rotated. The rotation can be 90, -90 and 180.

Returned result

If successful it returns 0.
If unsuccessful it returns an error code starting with $$ and the error code.

Special considerations

This can be used with the technology: QuickTimeClassic.
This function is currently only available on Mac OS X.

Example usage

Grab_SetRotation("-Unused" ; "QuickTimeClassic" ; 90)

This will rotate the preview image 90 degrees clockwise.

Grab_SetSettings

Syntax Grab_SetSettings(switches ; technologyName ; videoSettings)

This sets the video settings of the Grabber to the given videoSettings.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
technologyName	the technology to use
videoSettings	a video setting data you got as a result from "Grab_GetCurrentSettings"

Returned result

If successful it returns 0. If unsuccessful it returns an error code starting with $$ and the error code.

Special considerations

This can be used with the technology: QuickTimeClassic.
Important: Use only the binary data you got as a result from "Grab_GetCurrentSettings". This data is in a special format and can't be modified. If you want to use 2 (or more) different settings, store each specific setting returned from Grab_GetCurrentSettings in different containers.

Example usage

Set Field[errorCode, Grab_SetSettings("-Unused" ; "QuickTimeClassic" ; gGrabberSettingsContainer)]

Here gGrabberSettingsContainer is a container field, where you previously stored a setting with the Grab_GetCurrentSettings function. The contents of this container shows as a file with the name GrabberSettings.

Grab_SetWindowPosition

Syntax Grab_SetWindowPosition(switches ; left ; top)

Sets the position on the screen of the grabber dialog window to be shown.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
left	the left co-ordinate (in pixels) of the dialog
top	the top co-ordinate (in pixels) of the dialog

Returned result

An error code. Currently the plug-in always returns 0.

Special considerations

This can be used with the functions Grab_OpenImageGrabWindow and Grab_OpenMovieGrabWindow
This function is currently only available on Mac OS X.

Example usage

Set Field[gErrorCode, Grab_SetWindowPosition("-Unused" ; 100 ; 150)]

This will set the position of the next dialog box to 100 pixels from the left and 150 pixels from the top of the screen.

Example 2

This example will use fields to set the position the next dialog window. We assume that in your FileMaker file the following fields are defined:

gErrorCode	Global, text
gLeft	Global, text
gTop	Global, text

In ScriptMaker add the following script steps:

Set Field [gErrorCode, Dial_SetWindowPosition("-Unused " ; gLeft ; gTop)]

Grab_SetWindowSize

Syntax Grab_SetWindowSize(switches ; width ; height)

Sets the size of the grabber dialog window to be shown.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
width	the wanted width (in pixels) of the dialog
height	the wanted height (in pixels) of the dialog

Returned result

An error code. Currently the plug-in always returns 0.

Special considerations

This can be used with the functions Grab_OpenImageGrabWindow and Grab_OpenMovieGrabWindow
This function is currently only available on Mac OS X.

Example usage

Set Field[gErrorCode, Grab_SetWindowSize("-Unused" ; 500 ; 300)]

This will set the position of the next dialog box to 500 pixels wide and 300 pixels high.

Example 2

This example will use fields to set the position the next dialog window. We assume that in your FileMaker file the following fields are defined:

gErrorCode	Global, text
gWidth	Global, text
gHeight	Global, text

In ScriptMaker add the following script steps:

Set Field [gErrorCode, Dial_SetWindowSize("-Unused " ; gWidth ; gHeight)]

Grab_SetWindowTitle

Syntax Grab_SetWindowTitle(switches ; title)

Set the name of the grabber dialog window

Parameters

switches	these determine how this function works
title	the title to show at the top of the grabber dialog window

Returned result

Always returns 0

Special considerations

This can be used with the functions Grab_OpenImageGrabWindow and Grab_OpenMovieGrabWindow
This function is currently only available on Mac OS X.

Example usage

Grab_SetWindowTitle("-Unused" ; "My Video Solution")

Grab_StartPreview

Syntax Grab_StartPreview(switches ; technologyName; previewBounds)

This starts the video preview stream at the specified coordinates and dimensions.

Parameters

switches	(optional) these determine how this function works
technologyName	the technology to use
previewBounds	the rectangle where the preview must be displayed, relative to the window. Dimensions (in pixels) are in this order: "left top right bottom", which is the same order as the FileMaker FieldBounds function

switches can be left empty or can be this:

-MaintainProportions	this will keep the preview proportional
-EnableHighQuality	this will enable a high quality preview and recording on some video sources.

Returned result

If successful this function returns 0. If unsuccessful it returns an error code starting with $$ and the error code. Possible error codes are:

$$-2020 QuickTime not available
$$-4243 kErrNotImplemented, not available with this technology

Other errors may be returned.

Special considerations

This can be used with the technology: QuickTimeClassic.
Important: Use the function Grab_Initialise first to test if there is an input source available and to initialise it.
The results of Grab_Initialise are the maximum width and height. To get a proportional smaller image divide the width and height by the same factor.

Example usage

Set Variable [$result, Grab_StartPreview("-Unused" ; "QuickTimeClassic"; "20 120 100 180")]

This will start a preview at position left 20 and top 120, and right 100 and 180 bottom. So it is 80 pixels wide and 60 high.

Grab_StopPreview

Syntax Grab_StopPreview(switches ; technologyName)

This stops the video preview stream.

Parameters

switches	not used, reserved for future use. Leave blank or put "-Unused"
technologyName	the technology for which this is

Returned result

This function always returns 0.

Special considerations

This can be used with the technology: QuickTimeClassic.

Example usage

Set Field [result, Grab_StopPreview("-Unused"; "QuickTimeClassic")]

This will stop showing the preview.

Grab_Version

Syntax Grab_Version(switches)

Use this function to see which version of the plug-in is loaded.
Note: This function is also used to register the plug-in.

Parameters

switches *determine the behaviour of the function*

switches can be one of this:
-GetVersionString *the version string is returned (default)*
-GetVersionNumber *Returns the version number of the plug-in*
-ShowFlashDialog *Shows the Flash Dialog of the plug-in (returns 0)*
-GetPluginInstallPath *Returns the path where the plug-in is installed*

-GetRegistrationState *get the registration state of the plug-in: 0 = not registered ; 1 = registered*
-UnregisterPlugin *sets the registration state of the plug-in to unregistered*

If you leave the parameter empty the version string is returned.

Returned result

The function returns ? if this plug-in is not loaded. If the plug-in is loaded the result depends on the input parameter. It is either a:

VersionString:
If you asked for the version string it will return for example "Grabber Plug-in 2.3"

VersionNumber:
If you asked for the version number it returns the version number of the plug-in x1000. For example version 2.0 will return number 2000.

ShowFlashDialogResult:
This will show the flash dialog and then return the error code 0.

Special considerations

IMPORTANT Always use this function to determine if the plug-in is loaded. If the plug-in is not loaded use of external functions may result in data loss, as FileMaker will return an empty field to any external function that is not loaded.

Example usage

Grab_Version("") will return the current version of the plug-in, for example: "Grabber Plug-in 2.2".

Example 2

Grab_Version("-GetVersionNumber") will return 2000 for version 2.0.
Grab_Version("-GetVersionNumber") will return 1901 for version 1.9.0.1
Grab_Version("-GetVersionNumber") will return 2230 for version 2.2.3

So for example to use a feature introduced with version 2.0 test if the result is equal or greater than 2000.

Grab_VersionAutoUpdate

Syntax Grab_VersionAutoUpdate

Use this function to see which version of the plug-in is loaded, formatted for FileMaker Server's AutoUpdate function. Returns 8 digit number to represent an AutoUpdate version.

Parameters

none

Returned result

The function returns ? if this plug-in is not loaded. If the plug-in is loaded the result is a version number, it is returned in the format aabbccdd where every letter represents a digit of the level, so versions can be easily compared.

Special considerations

The Grab_VersionAutoUpdate function is part of an emerging standard for FileMaker plug-ins of third party vendors of plug-ins. The version number can be easily compared, when using the Autoupdate functionality of FileMaker Server.

Example usage

Grab_VersionAutoUpdate will return 02000100 for version 2.0.1

So for example to use a feature introduced with version 2.0 test if the result is equal or greater than 02000000.