Watching for Errors with MBS Plugin

When you use our MBS FileMaker Plugin, you will make a lot of calls to the plugin. Most will succeed, but some will fail. Do you notice when something fails?

IsError function

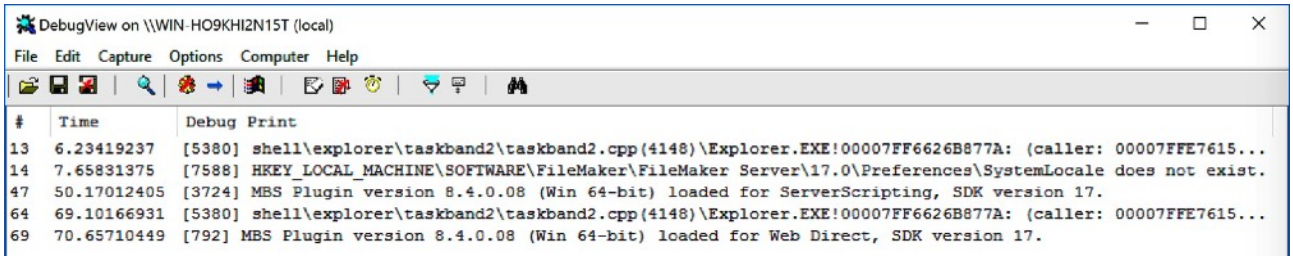
We see a lot of clients coding without checking for error state. But not all function calls succeed and our MBS FileMaker Plugin keeps a last error state for you. Even on server with multiple scripts running in parallel, we keep an error state per script. You can query it with MBS("IsError") to see if the last function call returned an error.

You may do an error check each time you call a function with an If script step and handle the script. You may even write a log entry in a text file (Text.AppendTextFile) or database (FM.InsertRecord). We have a nice custom function for this in this blog article to log Get(LastError) results, but you can adapt it to MBS FileMaker Plugin with using IsError function: Looping over records in FileMaker with error checking

Trace function

Every MBS Plugin user should know about the Trace function. This function allows you to write a log call of all MBS function calls made. This allows you to see your scripts run from the point of view of our plugin. You may see where you create, where you use and where you release objects. Once enabled, it stays active until you quit the current process, e.g. FileMaker Pro. Or you run Trace.Off function later.

If you pass a native file path to the Trace function, we create a log file there. But if you pass empty path, the log goes to the debug console. On macOS you can find it in Console.app, best by searching for MBS. On Windows you can use DebugView.exe from Microsoft. On a linux or macOS server the output goes to stderr log files in Logs folder. On Windows you can run DebugView.exe as admin, enable global Win32 listening and see live log messages in the window. Like in this screenshot:



You can configure trace with Trace.SetWithTimes function to include time stamps for each call. The plugin by default logs also evaluate calls

from the plugin as well as scripts triggered. Our plugin keeps an eye on what is current file and script names, so we can log if they change.

Here is a sample output:

```
Script "TraceTest" in file "Contacts".
MBS Plugin call #4 with 3 parameters at 17.03.2021 15:55:50.
Parameter 0: "Window.SetToolbarVisible"
Parameter 1: "test"
Parameter 2: 0
Result at 17.03.2021 15:55:50 #4: "[MBS] Invalid window
reference: test"
```

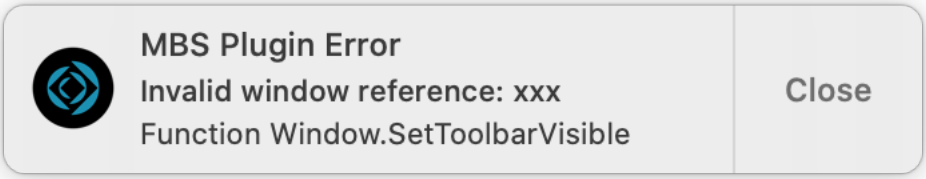
As you see it reports the script and file name. Then you see the call with a number, parameter count and timestamp. Next it lists parameters and later comes the result line. The result line shows als a timestamp and if the function takes longer, we also log the duration of the function call. If you have calls without result lines, that means a crash within the plugin and you should contact us, so we can debug the issue. Otherwise you the see the result here.

Numbers, dates, times and timestamps are shown as they are, text in quotes and containers list their content with types, sizes and a preview of data. But output is limited to a few hundred characters. You may use this information to verify you pass the right data. Too often people pass a field, but they are not on the right layout or record, so no value or the wrong one is passed.

If you don't want to log all calls, you could limit yourself to log failed calls. Use Trace.SetErrorsOnly with value 1 to do this. If everything works fine, your trace log file may stay very small in size.

On a server you can use Trace.SetServerLogPath function to put trace log files in a folder and have each script use a different file name. We use some numbers and reuse them, so you may end up with a handful of files with various scripts logging to them.

You may also use Trace.EnableErrorNotifications function to have macOS show notifications for errors in scripts. This is a bit filtered to avoid to many bubbles and not to repeat them. And we close them after a few seconds.



You can add your own log messages to our trace file with Trace.WriteLine function. e.g. have within a script log some additional data like record IDs.

Error Script Trigger

We got a feature where MBS FileMaker Plugin can trigger a script to report errors. Use Plugin.SetErrorScriptTrigger to set which script to call. We pass a JSON to the script as parameter with details:

- MBS function name
- Parameter values
- Result value
- UserName
- FileName
- ScriptName
- AccountName
- HostName
- RecordID
- CurrentTimeStamp
- LayoutTableName
- LayoutName
- WindowName
- TraceID

In the script you can then log the error into your database.

See Utility functions/Trigger Script on Plugin Error example database.

Don't miss your error handling

So please do something about catching your errors. A script that run fine on Friday may fail silently on the next Monday. Why? Because on Friday you had a network disk mounted. Over the weekend you restarted your computer and now on Monday your file paths are pointing to the non existing network share.

Please don't hesitate to contact us if you have questions.