1.5.0 Beta 11 LCFM Native ETS Release Notes

Release Notes

LCFM Native (1.5.0 Beta 11)

Tool

- The "Test & Deploy" button on the Projects screen has been renamed to "Deployment Options"
- A spinner has been added on FileMaker Plugin Update screen
- The correct field is now highlighted after choosing or dragging a solution.

Parsing

Valuelists

- Valuelists which cannot be edited in filemaker will no longer cause the parsing of a solution to fail.

Extracting

- Fields containing an invalid value will now be extracted correctly.
- Data from tables whose names contain non-ascii characters is now extracted correctly

Deployment

Android Deployment

- The status bar is now hidden correctly when the `Status Bar Hidden` checkbox is set.
- Apps no longer cause the application to exit when run on an android emulator.

iOS Deployment

- Deploying iOS apps from macOS 10.15.4 or later now requires Xcode 12.1 and uses iOS SDK 14.1.

Runtime

Find Mode

- There is now a request set for each table in a window, rather than each layout in a window.
- When entering find mode using `Enter Find Mode`, all request sets across all tables in the window are reset.

- Request sets now always have at least one request present when active.
- Queries entered into date, time and timestamp fields will now be validated correctly.

Functions

(Binary) Addition

- The result of the addition operator will now be of the correct type when adding with times, dates and timestamps.

(Binary) Subtraction

- The result of the subtraction operator will now be of the correct type when subtracting with times, dates and timestamps.

Extend

- The `Extend` function is now implemented.

Get(ActiveModifierKeys)

- The `Get(ActiveModifierKeys)` function is now implemented.

Get(ActiveSelectionSize)

- The `Get(ActiveSelectionSize)` function now returns the correct value on Android.

Get(ActiveSelectionStart)

- The `Get(ActiveSelectionStart)` function now returns the correct value on Android.

Get(ConnectionAttributes)

- The `Get(ConnectionAttributes)` function is now implemented.

Get(ConnectionState)

- The `Get(ConnectionState)` function is now implemented.

Get(HostIPAddress)

- The `Get(HostIPAddress)` function is now implemented.

Get(ModifiedFields)

- The `Get(ModifiedFields)` function is now implemented.

Get(MultiUserState)

- The `Get(MultiUserState)` function is now implemented.

Get(TriggerKeystroke)

- The `Get(TriggerKeystroke)` function is now implemented.

Get(TriggerModifierKeys)

- The `Get(TriggerModifierKeys)` function is now implemented.

Get(UseSystemFormatsState)

- The `Get(UseSystemFormatsState)` function is now implemented.

Get(WindowVisible)

- The `Get(WindowVisible)` function will now return true when the window is minimized.

GetAsDate

- The `GetAsDate` function now works correctly when its argument is formatted appropriately for the solution's locale

GetAsNumber

- The `GetAsNumber` function now works correctly when its argument is formatted appropriately for the solution's locale

GetAsText

- The `GetAsText` function now formats number, date, time and timestamp data correctly according to the solution's locale.

GetAsTime

- The `GetAsTime` function now works correctly when its argument is formatted appropriately for the solution's locale
- The `GetAsTime` function will now accept times with only a decimal separator and fractional component.

GetAsTimestamp

- The `GetAsTimestamp` function now works correctly when its argument is formatted appropriately for the solution's locale

GetLayoutObjectAttribute

- The `containedObjects` attribute for the `GetLayoutObjectAttribute` function has been implemented.

- The `enclosingObject` attribute for the `GetLayoutObjectAttribute` function will now return the correct value.

General

- Solution files with an initial layout that is unknown will no longer cause the application to exit on start.

Layout Objects

Entry Fields

- Entry fields now correctly indicate whether find mode is active.

Fields

- When fields are converted to text in calculations, the result will be correctly formatted according to the solution's locale.
- Numbers, times, dates and timestamps are now formatted correctly using either the system or solution's locale.
- Container fields containing legacy initial data will now display correctly as images when there is an alternative image stream present.
- Checkbox set fields using the classic theme now render correctly.
- Radio set fields using the classic theme now render correctly.
- Field objects with the legacy `engraved` style now render correctly.
- Entry fields no longer accept any input when the record is already being modified in a different window.
- Fields with no vertical scrollbar now add a vertical scrollbar when their content is larger than their maximum height allows.
- The insertion point color now matches the text color in entry fields on macOS
- Fields now support spell-checking and respect the `Do not apply visual spell-checking` option.
- Container field objects will now wrap their file name.
- Entering text using composition-based input method editing now works correctly on macOS.
- Container fields will no longer cause the application to exit when attempting to display an image with invalid image data.
- Entry fields will no longer be the incorrect height when focussed on in some cases.
- Changing a drop-down list or calendar field will no longer cause the application to exit if its `OnObjectModify` trigger changes the activ layout object.
- On android devices menus shown by container, drop-down list and popup menu fields are now themed correctly.
- Entry fields on Android no longer lose focus when the user taps other objects while they are active.
- The insert from camera menu option on container fields no longer causes the application to exit on Android 11.
- Entry fields in portal rows now scroll
- Global container fields containing image data will now be updated correctly when their value changes.
- Number fields using the `Decimal` formatting option with anything other than `.` as decimal separator will now be formatted correctly.

Popovers

- Clicking on a layout object in a popover will no longer close the popover.

Portals

- Portals being displayed in a window which is in find mode will now display the correct portal row.
- Objects in portal rows now sit inside the border of the row, rather than on top of it.

Tab Controls

- Tab controls and their panels now behave correctly when the layout is scrolled.
- Tab controls and their panels now behave correctly when the window is resized.
- Layouts with tab controls on them no longer cause the application to exit when in list view.

Text

- Text objects from older solutions which should expand to fit their content rather than wrap now do so.

Layouts

- Fixed parts now update when the active record changes in list view mode.

Relationships

- Relationships are now computed correctly when a source field of number type contains carriage returns
- Empty and invalid values will no longer cause a match in number, date, time and timestamp fields.

Script Steps

Adjust Window

- The `Minimize` option for the `Adjust Window` script step will now hide the window on mobile.

Delete Portal Row

- Warning dialogs displayed as a result of the `Delete Portal Row` script step include have a warning icon

Delete Record/Request

- Warning dialogs displayed as a result of the `Delete Record/Request` script step now include a warning icon

Exit Script

- The `Exit Script` step now returns the specified script result correctly when executed whilst a script is paused.

Go To Related Record

- The `Go To Related Record` script step now shows the correct records.
- The 'Use external table's layouts' option of the `Go To Related Record` script step is now honored

Import Records

- Warning dialogs displayed as a result of the `Import Records` script step now include a warning icon

Insert From Device

- On android devices, when using the `Insert From Device` script step, the menu is now themed correctly.
- The `Insert From Device` script step in `Camera` mode no longer causes the application to exit on Android 11.

Insert From URL

- The `Insert From URL` script step now returns the appropriate error code if the downloaded data cannot be converted to text, but has been requested as text.

Move/Resize Window

- The `Move/Resize Window` script step will now make hidden windows visible if a size or position is specified.

Pause Script

- The `Pause Script` script step now behaves correctly when it is run from a button inside a portal row.

Print

- Printing a blank record as formatted, with blank boxes, with placeholders and underlined is now implemented
- Gradient fill styles while printing are now implemented
- Printing grouped objects is now implemented
- The evaluation of conditional formatting while printing is now implemented

Save Records As PDF

- Printing a blank record as formatted, with blank boxes, with placeholders and underlined is now implemented
- Gradient fill styles while printing are now implemented
- Printing grouped objects is now implemented
- The evaluation of conditional formatting while printing is now implemented

Set Field

- The `Set Field` script step now throws the correct errors.

Set Multi-User

- The `Set Multi-User` script step has now been implemented.

Set Use System Formats

- The `Set Use System Formats` script step is now implemented.

Set Variable

- The `Set Variable` script step will now treat a repetition evaluating to empty as repetition 1.

Show Custom Dialog

- Using the `Show Custom Dialog` script step immediately after closing the active window will no longer cause the application to exit or mobile devices.

Sync

- Local data which remains unsynced when the application closes will now be synced when the application is relaunched.
- Changes which have been retrieved from the server but not applied locally when the application closes will now be applied when the application is relaunched.
- Server timestamps are now adjusted to the client system clock before resolving conflicts in records which have been modified remotely and locally.
- Syncs where deletion checks are not performed no longer fail to push local data to the server when the primary key field name has spaces in.
- Tables whose names contain non-ascii characters now sync correctly
- Container data is now fetched correctly when syncing solutions hosted on FileMaker Server 19

Tables

- The `summarize repetitions individually` option for summary fields has now been implemented.
- The `Do not evaluate if all referenced fields are empty` option for calculated fields is now implemented
- The `Do not evaluate if all referenced fields are empty` option for field auto-enter calculations is now implemented
- Evaluation of repeated fields from a field calculation will now return the correct repetition.
- Changing a field which causes an auto-enter calculation on a repeated field to update will now operate on the correct repetition.
- Changing a field which causes a stored calculated field to update will now either update the correct (single) repetition or all repetitions, depending on how the field was referenced.
- Error dialogs displayed as a result of failed field validation now include an error icon
- Create through relationship now correctly cascades backwards through the relationship chain as far as necessary. This means it can now correctly succeed even when some intermediate joins do not admit creation.
- Indexed text fields with extracted data now work correctly regardless of language and content.
- Missing fields without an explicit repetition will now evaluate as empty.

Triggers

- The `OnObjectKeystroke` trigger is now implemented.
- The `OnLayoutKeystroke` trigger is now implemented.

LCFM Native (1.5.0 Beta 9)

Tool

- The main window of the tool is now centered on screen

Extracting

- Attempting to extract data from records where external container data is not available will no longer cause building to fail.

Runtime

Custom Functions

- Evaluating a calculation containing a reference to a deleted custom function will no longer cause the application to exit.

Find Requests

- Find Request multi-value criteria (eg =a =b) now check both values for a match when the target field is unindexed.

Functions

Get(WindowContentHeight)

- The `Get(WindowContentHeight)` function will now return the correct value for card windows

Get(WindowContentWidth)

- The `Get(WindowContentWidth)` function will now return the correct value for card windows

Get(WindowHeight)

- The `Get(WindowHeight)` function will now return the correct value for card windows

Get(WindowLeft)

- The `Get(WindowLeft)` function will now return the correct value for card windows

Get(WindowTop)

- The `Get(WindowTop)` function will now return the correct value for card windows

Get(WindowWidth)

- The `Get(WindowWidth)` function will now return the correct value for card windows

GetLayoutObjectAttribute

- The `GetLayoutObjectAttribute` function will now return empty if the object name is empty.

GetSummary

- The `GetSummary` function will now only return a value if the foundset is sorted by the break field.

RelationInfo

- The `RelationInfo` function will now return fully qualified field names with the correct field name.

TextDecode

- The `TextDecode` function no longer causes the application to exit when decoding text using the `ISO-8859-1` character set on iOS and MacOS and the `Macintosh` character set on Android.

TextEncode

- The `TextEncode` function no longer causes the application to exit when encoding text using the `ISO-8859-1` character set on iOS and MacOS and the `Macintosh` character set on Android.

Layout

- Flinging to scroll on Android now works correctly.

Layout Objects

Fields

- Container fields with the `Interactive content` formatting option turned on will no longer be cleared when pressing backspace.
- The active field will now scroll into view when the mobile keyboard is shown in list view.
- Newlines are no longer entered into android fields if the field goes to the next object using return.
- The mobile keyboard now shows a next style button if the field exits on return.
- Fields with the `Exit on return` option enabled no longer skip the next field when pressing return on some Android devices.
- Choosing the `Export Field Contents` option in a container field's context menu will no longer cause the application to exit.
- Calculations involving the current active field will now work correctly when it its datatype is date, time or timestamp.
- Displaying field objects containing a time with a fractional part using either `Short System Time` or `Long System Time` formatting will no longer cause the application to exit.
- Container fields with the `Interactive content` formatting option turned on will now be able to scroll their content.
- Styled text from the original solutions will now display correctly in fields.
- The text color is now set correctly on iOS concealed entry fields
- Placeholders are now correctly positioned when the field is active

Portals

- Related fields in portals will now display the correct value when they are related to the portal's table.

Slide Controls

- Scrolling now works correctly on mobile after swiping on a slide control.

Relationships

- Relationship conditions which use the same field more than once will now work correctly.

Script Steps

Commit Records/Requests

- The `Commit Records/Requests` script step will now cause new records to be added to all unfiltered foundsets sharing the same basetable.
- The `Commit Records/Requests` script step will now ensure any deleted related records are removed from all foundsets sharing the same basetable.
- The `Commit Records/Requests` script step will now ensure any deleted related records are removed from all relatedsets sharing the same basetable.
- The `Commit Records/Requests` script step will now ensure any deleted records are removed from all relatedsets sharing the same basetable.

Delete All Records

- The `Delete All Records` script step will now show all records after deleting the records of a filtered foundset.

Go To Layout

- The `by name` variant of `Go To Layout` will no longer cause the application to exit if there are multiple layouts with the same name.

Go to Object

- The `Go to Object` script step will now set the error code to 5 (invalid command) if the object name is empty.

Go To Related Record

- The `Go To Related Record` script step will now return the correct error when no records related to the current record are found
- The `Go To Related Record` script step will now go to the correct record when it is used from an active row in a filtered or sorted portal
- The `Go To Related Record` script step no longer causes the application to exit after being used when the current foundset is empt

Insert From Url

- The `Insert From Url` script step no longer causes the application to exit when fetching URLs encoded using the `ISO-8859-1` character set on iOS and MacOS and the `Macintosh` character set on Android.

Omit Multiple Records

- The `Omit Multiple Records` script step has now been implemented.

Omit Records

- The `Omit Records` script step has now been implemented.

Perform Script

- The `by name` variant of `Perform Script` will no longer cause the application to exit if there are multiple scripts with the same name
- The 'by name' variant of `Perform Script` will no longer cause the application to exit.

Print

- The `Print` script step will now return the correct error code on Android if the user cancels printing.

Refresh Object

- The `Refresh Object` script step will now set the error code to 116 (object is missing) if the object name is empty or the object is not visible.
- The `Refresh Object` script step no longer cause the application to exit when the target object's owning part is not currently visible.
- The `Refresh Object` script step no longer causes the application to exit when targeting an object on a popover in the popover's `OnObjectEnter` trigger.

Refresh Portal

- The `Refresh Portal` script step will now set the error code to 116 (object is missing) if the object name is empty.
- The `Refresh Portal` script step no longer causes the application to exit when targeting a portal on a popover in the in the popover's `OnObjectEnter` trigger.

Set Field

- The `Set Field` script step now converts values stored into time, date and timestamp fields correctly.

Set Web Viewer

- The `Set Web Viewer` script step will now set the error code to 5 (invalid command) if the object name is empty.

Sort Records

- The `Sort Records` script step option `Keep records in sorted order` is now implemented.

Sort Records by Field

- The `Sort Records by Field` script step now ensures records are kept in sorted order.
- The `Sort Records By Field` script step will no longer cause the application to exit.

Sync

- Empty container fields now sync correctly.

Tables

- Summary field evaluation now takes into account whether the field is being evaluated inside of a grand or sub summary part - changing the computation appropriately.
- Date fields are now indexed correctly.
- Time fields are now indexed correctly.
- Timestamp fields are now indexed correctly.
- `Total of` summary fields will now be evaluated correctly.
- Evaluation of `weighted average` summary fields is now implemented.

Windows

- The positions of touches in card windows are now calculated correctly on Android

LCFM Native (1.5.0 Beta 10)

Tool

- The tool no longer exits when clicking the "Learn about these settings" link in the Sync pane of the Advanced Settings dialog.

Runtime

Find Requests

- Find requests checking for empty or non-empty fields no longer cause the application to exit.

Functions

SortValues

- The `SortValues` function is now implemented.

UniqueValues

- The `UniqueValues` function is now implemented.

Layout Objects

Fields

- Container field content now displays correctly when the style or bounds of the field changes.
- The width of drop-down list field popups are no longer restricted to that of their field.
- Tapping repeatedly on drop-down list or calendar fields on iOS no longer causes the application to become unresponsive.
- The height of fields with vertical scrollbars set to never now resets to the content height when the width of the field is changed.

Layouts

- Sub-summary parts will not be added for new uncommitted records.
- Sub-summary parts will now be correctly placed when there is an active sort with multiple criteria.

Script Steps

Adjust Window

- The `Maximize` and `Minimize` options for the `Adjust Window` script step are now implemented.

Duplicate Record/Request

- The `Duplicate Record/Request` script step will now place new records correctly if there is an active sort on the foundset.

Export Records

- Using the `Export Records` script step to export data in comma and tab separated text format has been implemented.

Go To Object

- The `Go To Object` script step now works correctly when targeting tab and slide panels.

Import Records

- Using the `Import Records` script step to import data from text files in tab, comma or custom separated value format has been implemented.

New Record/Request

- The `New Record/Request` script step will now place new records correctly if there is an active sort on the foundset.

Print

- The `Delineate fields on current record only` layout option is now implemented when printing.
- Printing image objects is now implemented
- The evaluation of hide conditions while printing is now implemented
- The `Print` script step will no longer cause the application to exit if there are objects which have not had certain text styles set.

Save Records As PDF

- The `Delineate fields on current record only` layout option is now implemented when printing.
- Printing image objects is now implemented
- The evaluation of hide conditions while printing is now implemented

- The `Save Records As PDF` script step will no longer cause the application to exit if there are objects which have not had certain text styles set.

Feature Support

Files

Files and their basic commands and functions are implemented.

Each file in a compiled lcfm-native app corresponds to one solution file that was included the lcfm-native project.

Files can reference other files, but the name used for the reference must always be the leaf (without extension) name of the solution file included in the app.

Each file can create multiple windows, and database access and script execution works across files.

User accounts are not currently implemented. The User Name is always 'System User' and the account name is always '[Full Access]'.

Locales other than en_GB and en_US are yet to be implemented.

The OnFileAVPlayerChanged trigger is yet to be implemented.

Menusets are yet to be implemented.

Accounts, privilege sets and extended privileges are yet to be implemented.

Layout Objects

The majority of layout objects are implemented.

Layout object tooltips are not yet implemented.

The following layout object triggers are not yet implemented:

- OnObjectAVPlayerChange

Button

Button layout objects are implemented.

Button Bar

Button bar layout objects are implemented.

Button Bar Button

Button bar button layout objects are implemented.

Button Bar Popover Button

Button bar popover button layout objects are implemented.

Calendar Field

Calendar field layout objects are implemented.

Autocomplete is not currently supported.

Check Box Field

Check Box field layout objects are implemented.

The Allow Other option will include a check box for 'Other...' but its action is not yet implemented.

Container Field

Container field layout objects are implemented.

The container field layout object supports GIF, JPEG, PNG and BMP format images.

The container field layout object supports MP3, MP4 and WAV audio and video formats.

The container field layout object supports PDF files, although currently only the first page is shown and they are not pannable.

The container field layout object supports general file content.

Edit Box Field

Edit field layout objects are implemented.

Autocomplete is not currently supported.

Group Button

Group button layout objects are implemented.

Currently children of a group button object will not highlight whilst the mouse is pressed on the group button.

Image

Image layout objects are implemented.

Line

Line layout objects are implemented.

List Field

List field layout objects are implemented.

The Autocomplete option is not yet implemented.

The Allow Edit option will cause 'Edit...' to be added to the list, but its action is not yet implemented.

Note: List field objects which were created in historic versions of FileMaker and not changed in more recent versions can loose their 'Show Icon property. To rectify this, toggle the property value in the FileMaker UI and re-run the import phase.

Menu Field

Menu field layout objects are implemented.

The Allow Edit option will cause 'Edit...' to be added to the list, but its action is not yet implemented.

The Allow Other option will cause 'Other...' to be added to the list, but its action is not yet implemented.

Note: Menu field objects which were created in historic versions of FileMaker and not changed in more recent versions can loose their 'Show Icon' property. To rectify this, toggle the property value in the FileMaker UI and re-run the import phase.

Oval

Oval layout objects are implemented.

Popover

Popover layout objects are implemented.

Popover Button

Popover button layout objects are implemented.

Portal

Portals are implemented.

The 'retain scrollbar position' option is not yet implemented.

The 'allow row deletion' option is not yet implemented.

Radio Field

Radio field layout objects are implemented.

The Allow Other option will include a check box for 'Other...' but its action is not yet implemented.

Rectangle

Rectangle layout objects are implemented.

Rounded Rectangle

Rounded rectangle layout objects are implemented.

Slide Control

Slide control layout objects are implemented.

Using a swipe gesture to manipulate a slide control is not yet implemented.

Tab Control

Tab control layout objects are implemented.

Text

Text layout objects are implemented.

Webviewer

Webviewer layout objects are implemented.

Custom url schemes are supported, and you can specify the schemes which should be handled at runtime when compiling a project. Only solution files compiled into the app can be used. Use either the `$` or `~` form of custom urls, and use the file name of the solution as provided t the compiler.

Not Yet Implemented

- Chart

Valuelists

Valuelists are almost completely implemented.

The order and uniqueness of values in non-custom valuelists is determined by sorting case insensitively using standard Unicode rules. Using the index language or the resort-language setting is yet to be implemented.

Layouts

Layouts are implemented.

The following layout triggers are not yet implemented:

- OnGestureTap
- OnExternalCommandReceived

Preview mode is not yet implemented.

Table view state is not yet implemented.

The status bar, formatting bar and ruler are not yet supported.

Steps (and scripts) execute as if Error Capture is turned On. Automatic error handling when error capture is off and the dialogs which appear a a result are yet to be implemented.

Automatic scrolling into view of the active layout object is not yet implemented.

Typing into the active field whilst it is not visible is not yet implemented.

On macOS, if the system preference 'always show scrollbars' option is enabled then the scrolling content area size is not yet correct.

Layout mode is not supported.

JSON Implementation Details

In order to perform the JSON functions the JSON is decoded and then encoded which results in some interpretation of values when re-encoding:

- Empty objects or arrays will be empty strings in the result.
- String values `"true"` and `"false"` will be evaluated as booleans and therefore will be `true` or `false` in the result. Where they ar returned as scalar values they will be the numbers `0` and `1`.
- String value `"null"` will be evaluated as `null` and therefore will be `null` in the result. Where it is returned as a scalar value it will b an empty string or in the case of `JSONListValues` it will not be included in the list.
- String values that are decimal numbers will be numbers in the result.
- Objects that have sequential integer keys starting from 1 will be arrays in the result. For example `{ "1" : true, "2", false }` will be returned as `[ true, false ]`.

Layout Parts

Layout parts are implemented.

Printing Implementation Details

Printing of simple print-oriented layouts is implemented.

The `Enter Preview Mode` script step works on all platforms, and has an initial prototype user interface allowing exit, page navigation, saving as pdf, page setup and print.

The `Print` script step works on all platforms and allows printing a layout through the system printing user interface.

The print-related part options are implemented.

The `Do not print`, `Sliding left` and `Sliding up` print-related layout object options are implemented.

Printing of the following layout objects is implemented:

- Field objects (only one repetition is currently printed)
- Rectangle objects
- Rounded Rectangle objects
- Grouped objects
- Oval objects
- Line objects
- Image objects

Printing a blank record as formatted, with blank boxes, with placeholders and underlined is implemented.

The `Delineate fields on current record only` layout option is implemented.

The evaluation of hide conditions while printing is implemented.

The evaluation of conditional formatting while printing is implemented.

The majority of style related options is implemented. Notable exceptions are:

- full justification of text
- text tab stops
- image fills

Tables, Fields and Relationships

Tables and fields are almost completely implemented.

Field validation is yet to be implemented.

The override language option for sorting is yet to be implemented.

Table, record and field level access are yet to be implemented.

Find Request Query Operators

Find request query operators are partially implemented.

The following operators are available:

- is empty: `=`
- is not empty: `*`
- is less than or equal: `<=<value>` or `≤<value>`
- is greater than or equal: `>=<value>` or `≥<value>`
- is less than: `<<value>`
- is greater than: `><value>`
- ranges: `<value>..<value>` or `<value>...<value>`
- is equal to: `<value>` or `=<value>` or `==<value>` or `"<value>"` (when the datatype is not text)
- is exactly: `==<value>` (when the datatype is text)
- contains word: `=<value>` (when the datatype is text)
- contains word beginning with: `<value>` (when the datatype is text)
- contains phrase beginning with: `"<value>"` (when the datatype is text)

Find request criteria using `>`, `>=`, `≥`, `<`, `<=` and `≤` may have spaces between the operator and value.

Functions

Not Currently Supported

Context

GetAllowFormattingBarState

Not currently supported. Returns 0.

GetStatusAreaState

Not currently supported. Returns 0.

GetTextRulerVisible

Not currently supported. Returns 0.

Implemented With Differences

Container

CryptAuthCode

Implemented with differences. The SHA and MDC2 digest algorithms are not supported.

CryptDigest

Implemented with differences. The SHA and MDC2 digest algorithms are not supported.

Context

GetAccountExtendedPrivileges

Implemented as override. Currently the value returned by Get(AccountExtendedPrivileges) will be that specified by the corresponding override at compile-time, or empty if one isn't set.

GetAccountGroupName

Implemented as override. Currently the value returned by Get(AccountGroupName) will be that specified by the corresponding override at compile-time, or empty if one isn't set.

GetAccountName

Implemented as override. Currently the value returned by Get(AccountName) will be that specified by the corresponding override at compile-time, or

GetAccountPrivilegeSetName

Implemented as override. Currently the value returned by Get(AccountPrivilegeSetName) will be that specified by the corresponding override a compile-time, or "[Full Access]" if one isn't set.

GetAccountType

Implemented as override. Currently the value returned by Get(AccountType) will be that specified by the corresponding override at compile-time, or "FileMaker File" if one isn't set.

GetApplicationVersion

Implemented with override. The value returned by Get(ApplicationVersion) will be that specified as the corresponding override at compile-time. If no override is set then "Go 18.0.0" is returned.

GetConnectionAttributes

Implemented with differences. Solutions run locally and are not hosted so the return value is always empty.

GetConnectionState

Implemented with differences. Solutions run locally and are not hosted so the return value is always 0.

GetDevice

Implemented with override. The value returned by Get(Device) will be that specified as the corresponding override at compile-time. If no override is set then it returns the same value as FileMaker for the current platform.

GetHostIPAddress

Implemented with differences. Solutions run locally and are not hosted so the return value is always empty.

GetMultiUserState

Implemented with differences. Solutions run locally and are not hosted so the return value is always 0.

GetPersistentID

Implemented with differences. The id returned is generated on first-run of the app on a given device. The id will survive backups but not necessarily deletion and reinstallation of the app.

GetSystemPlatform

Implemented with override. The value returned by Get(SystemPlatform) will be that specified as the corresponding override at compile-time. If no override is set then it returns the same value as FileMaker for the current platform.

GetUserName

Implemented with override. The value returned by Get(UserName) will be that specified as the corresponding override at compile-time. If no override is set then it returns the full name of the current user on macOS, `iPad` or `iPhone` on iOS, and `Android` on android.

GetWindowOrientation

Implemented with override. The value returned by Get(WindowOrientation) will be that specified as the corresponding override at compile-time If no override is set then it returns the same value as FileMaker would.

Design

WindowNames

Implemented with differences. The list of window names for a given file are in order of most recently opened or brought to the front. If no file name is specified then the windows are grouped by file in the list.

Logical

Evaluate

Implemented with differences. If the expression string fails to parse then the evaluation error will be set to 1200 (generic calculation error), rather than a more specific code.

Mobile

Location

Implemented with differences. Implemented on mobile platforms only. Allowing users to cancel the location fetching process when Get(AllowAbortState) is true is yet to be implemented.

LocationValues

Implemented with differences. Implemented on mobile platforms only. Allowing users to cancel the location fetching process when Get(AllowAbortState) is true is yet to be implemented.

Text

JSONDeleteElement

Implemented with differences. See the `JSON Implementation Details` section for details.

JSONFormatElements

Implemented with differences. See the `JSON Implementation Details` section for details.

JSONGetElement

Implemented with differences. See the `JSON Implementation Details` section for details.

JSONListKeys

Implemented with differences. See the `JSON Implementation Details` section for details.

JSONListValues

Implemented with differences. See the `JSON Implementation Details` section for details.

JSONSetElement

Implemented with differences. See the `JSON Implementation Details` section for details.

Not Fully Implemented

Container

GetContainerAttribute

Not fully implemented. Only the following attributes can be used:

- filename
- barcodeText
- barcodeType

Logical

GetLayoutObjectAttribute

Not fully implemented. Only the following attributes can be used:

- objectType
- hasFocus
- isFrontPanel
- isFrontTabPanel

- isObjectHidden
- containedObjects
- rotation

IsValid

Not fully implemented. The ability to check whether a field's content matches its data-type is not yet implemented.

Not Yet Implemented

Aggregate

- Last

Container

- GetHeight
- GetWidth
- VerifyContainer

Context

- GetApplicationLanguage
- GetCurrentExtendedPrivileges
- GetCurrentMenuSetName
- GetCurrentPrivilegeSetName
- GetDocumentsPathListing
- GetEncryptionState
- GetFileMakerPath
- GetHighConstrastState
- GetHighContrastColor
- GetHostApplicationVersion
- GetHostName
- GetInstalledFMPlugins
- GetLayoutAccess
- GetMenubarState
- GetNetworkProtocol
- GetNetworkType
- GetPreferencesPath
- GetPrinterName
- GetQuickFindText
- GetRecordAccess
- GetRegionMonitorEvents
- GetScreenDepth

- GetScreenScaleFactor
- GetScriptAnimationState
- GetSystemDrive
- GetSystemIPAddress
- GetSystemLanguage
- GetSystemNICAddress
- GetTouchKeyboardState
- GetTriggerExternalEvent
- GetTriggerGestureInfo
- GetUserCount
- GetWindowZoomLevel

Design

- FieldBounds
- FieldIDs
- FieldRepetitions
- FieldStyle

Logical

- LookupNext

Mobile

- GetSensor
- RangeBeacons

Number

- Combination
- Factorial

Text

- Furigana
- Hiragana
- KanaHankaku
- KanaZenkaku
- KanjiNumeral
- Katakana
- NumToJText
- RomanHankaku
- RomanZenkaku

Not Supported

Container

- CryptDecrypt
- CryptDecryptBase64
- CryptEncrypt
- CryptEncryptBase64

Script Steps

Not Currently Supported

Miscellaneous

Allow Formatting Bar

Not currently supported. Has no effect.

Windows

Show Hide Text Ruler

Not currently supported. Has no effect.

Show Hide Toolbars

Not currently supported. Has no effect.

Remote operations

A number of script steps can only be run on the server. Currently this collection consists of the account-related script steps and Perform Script On Server. The steps are executed by running particular scripts on the server after logging in via the data API. As such, in order for them to work, there are some application prerequisites:

1. The solution must be hosted on FileMaker Server
2. Any account under which the script steps are intended to be run must have the fmrest privilege
3. The target script must exist in the hosted solution.

   Important: As they are executed via the Data API, the bespoke scripts must be uniquely named, otherwise the correct script may not be called, even if the scripts are in different folders.

In the case of the account-related script steps, with the exception of Re-Login, the scripts in question should be of a specific form:

```
# LCFM Native <Operation Name> Script

Set Error Capture [ On ]
<Run account script step>
Exit Script [ Text Result: Get(LastError) ]
```

Examples for each step can be seen below.

Add Account

The Add Account remote script must contain the comment

```
# LCFM Native Add Account Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Expire password
2. Account name
3. Password
4. Privilege set name

The implementation itself should not use the 'User must change password on next sign-in' (Expire password) option, as it is not supported. Since the privilege set cannot be a calculation, each case must be hard-coded in an If ... Else If ... block.

Example:

```
# LCFM Native Add Account Script

Set Error Capture [ On ]

# The expire password variable is not used. If it is true, return
# "invalid-command".
Set Variable [ $expire_password ; Value: GetValue(Get(ScriptParameter);1) ]
If [ $expire_password ]
    Exit Script [ Text Result: 3 ]
End If

Set Variable [ $username ; Value: GetValue(Get(ScriptParameter);2) ]
Set Variable [ $password ; Value: GetValue(Get(ScriptParameter);3) ]
Set Variable [ $privilege_set ; Value: GetValue(Get(ScriptParameter);4) ]

If [ $privilege_set = "[Data Entry Only]" ]
    Add Account [ Account Name: $username ; Password : $password ;
    Privilege Set: [Data Entry Only] ]
Else If [ $privilege_set = "[Read-Only Access]" ]
    Add Account [ Account Name: $username ; Password : $password ;
    Privilege Set: [Read-Only Access] ]
Else
    # Handle custom privilege sets
    If [ $privilege_set = "My Custom Privilege Set" ]
        Add Account [ Account Name: $username ; Password : $password ;
        Privilege Set: My Custom Privilege Set ]
    Else
        # Privilege set does not exist
        Exit Script [ Text Result: 3 ]
    End if
End If
Exit Script [ Text Result: Get(LastError) ]
```

Change Password

The Change Password remote script must contain the comment

```
# LCFM Native Change Password Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Current password
2. New password

Example:

```
# LCFM Native Change Password Script

Set Error Capture [ On ]

Set Variable [ $old_password ; Value: GetValue(Get(ScriptParameter);1) ]
Set Variable [ $new_password ; Value: GetValue(Get(ScriptParameter);2) ]

Change Password [ Old Password: $old_password ; Password : $new_password
With dialog: Off ]
Exit Script [ Text Result: Get(LastError) ]
```

Delete Account

The Delete Account remote script must contain the comment

```
# LCFM Native Delete Account Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Account name

Example:

```
# LCFM Native Delete Account Script

Set Error Capture [ On ]

Set Variable [ $account_name ; Value: GetValue(Get(ScriptParameter);1) ]

Delete Account [ Account Name: $account_name ]
Exit Script [ Text Result: Get(LastError) ]
```

Enable Account

The Enable Account remote script must contain the comment

```
# LCFM Native Enable Account Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Activate (true or false)
2. Account name

Since whether to activate or deactivate cannot be a calculation, each case must be hard-coded in an If ... Else ... block.

Example:

```
# LCFM Native Enable Account Script

Set Error Capture [ On ]

Set Variable [ $is_activate ; Value: GetValue(Get(ScriptParameter);1) ]
Set Variable [ $account_name ; Value: GetValue(Get(ScriptParameter);2) ]

If [ $is_activate ]
    Enable Account [ Account Name: $account_name ; Activate ]
Else
    Enable Account [ Account Name: $account_name ; Deactivate ]
End If
Exit Script [ Text Result: Get(LastError) ]
```

Re-Login

The Re-Login script step can be run without a remote script implementing it. In this case, the credentials are simply used to authorize Data API login. The script step can thus be used to verify that the account exists and the password is correct. In this case no local state is changed.

A remote script for Re-Login is needed in order to have the account related Get functions behave correctly after relogin.

The Re-Login remote script must contain the comment

```
# LCFM Native Re-Login Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Account name
2. Password

The Re-Login remote script *must* return the account data as a carriage-return-delimited list in the correct order:

1. Last error code
2. Account type
3. Account name
4. Account group name
5. Account privilege set name
6. Account extended privileges

> *Important*: If the text result of this script differs from the above-specified format, the application may subsequently exhibit undesired behavior.

> *Note*: Since the provided credentials are used to log in to the Data API, it is not necessary for the Re-Login script to actually run the Re-Login script step. The key function of the script is to return the relevant account data.

Example:

```
# LCFM Native Re-Login Script

Exit Script [ Text Result: 0 & ¶ & Get(AccountType)
     & ¶ & Get(AccountName) & ¶ & Get(AccountGroupName)
     & ¶ & Get(AccountPrivilegeSetName) & ¶ & Get(AccountExtendedPrivileges) ]
```

Reset Account Password

The Reset Account Password remote script must contain the comment

```
# LCFM Native Reset Account Password Script
```

somewhere within the script. The parameters passed when executing this script are:

1. Expire password
2. Account name
3. Password

The implementation itself should not use the 'User must change password on next sign-in' (Expire password) option, as it is not supported.

Example:

```
# LCFM Native Reset Account Password Script

Set Error Capture [ On ]

# The expire password variable is not used. If it is true, return
# "invalid-command".
Set Variable [ $expire_password ; Value: GetValue(Get(ScriptParameter);1) ]
If [ $expire_password ]
    Exit Script [ Text Result: 3 ]
End If

Set Variable [ $account_name ; Value: GetValue(Get(ScriptParameter);2) ]
Set Variable [ $password ; Value: GetValue(Get(ScriptParameter);3) ]

Reset Account Password [ Account Name: $account_name ; Password : $password ]
Exit Script [ Text Result: Get(LastError) ]
```

Implemented

Fields

Insert From Device Bar Code From Camera

Implemented on mobile platforms only.

Insert From Device Music Library

Implemented on mobile platforms only.

Insert From Device Photo Library

Implemented on mobile platforms only.

Implemented With Differences

Account

Add Account

Implemented with differences. This script step can only be run on the server. See the 'Remote Operations' section for details. The 'User must change password on next sign-in' option is not supported.

Change Password

Implemented with differences. The 'with dialog' form is not yet implemented. This script step can only be run on the server. See the 'Remote Operations' section for details.

Delete Account

Implemented with differences. This script step can only be run on the server. See the 'Remote Operations' section for details.

Enable Account

Implemented with differences. This script step can only be run on the server. See the 'Remote Operations' section for details.

Re Login

Implemented with differences. The 'with dialog' form is not yet implemented. This script step can only be run on the server. See the 'Remote Operations' section for details.

Reset Account Password

Implemented with differences. This script step can only be run on the server. See the 'Remote Operations' section for details. The 'User must change password on next sign-in' option is not supported.

Control

Allow User Abort

Implemented with differences. The AllowUserAbort state can be set and retrieved with Get(AllowAbortState), but all script steps run as if it is se to False.

Set Error Capture

Implemented with differences. The ErrorCapture state can be set and retrieved with Get(ErrorCaptureState), but all script steps run as if it is se to True.

Fields

Export Field Contents

Implemented with differences. On desktop platforms, the 'Auto Open' and 'Create Email' options are not available.

Export Field Contents To File

Implemented with differences. On desktop platforms, the 'Create Email' option has no effect.

Insert From Device Bar Code From Field

Implemented with differences. Implemented on android platform only.

Files

Close File

Implemented with differences. Closing a file in lcfm-native causes all its windows and reverts it to being in the background.

Open File

Implemented with differences. All files compiled into an lcfm-native app are opened in the background on startup, calling Open File causes the file to be brought to the foreground, running appropriate triggers.

Note: You can only open files which were included in the app at compile-time.

Set Multi User

Implemented with differences. Solutions run locally and cannot be shared so this has no effect.

Miscellaneous

Configure Local Notification

Implemented with differences. Buttons on the notifications are not yet supported and in the script parameter line two will always be `NotificationSelected` and line 3 will always be 0 as these values are not yet being determined.

Records

Commit Records Requests

Implemented with differences. The skip-validation option currently has no effect as field validation is not yet implemented. The 'with dialog' form is not yet implemented. The force-commit option is not supported.

Windows

Refresh Window

Implemented with differences. The flush-joins option is currently has no effect as the join cache is automatically flushed more frequently. The flush-sql option is not supported.

View As

Implemented with differences. The table view mode is not supported.

Not Fully Implemented

ExportRecords

Export Records

Not fully implemented. Only exporting without dialog to tab or comma-separated values is implemented. Exporting to FileMaker files is not supported.

Fields

Replace Field Contents With Calculation

Not fully implemented. The 'with dialog' form is not yet implemented.

Replace Field Contents With Current Contents

Not fully implemented. The 'with dialog' form is not yet implemented.

Replace Field Contents With Serial Numbers

Not fully implemented. The 'with dialog' form is not yet implemented.

Foundsets

Omit Multiple Records

Not fully implemented. The 'with dialog' form is not yet implemented.

Sort Records

Not fully implemented. The 'with dialog' form is not yet implemented.

ImportRecords

Import Records

Not fully implemented. Only importing into existing tables without dialog from tab or comma separated values is implemented. Importing from FileMaker files is not supported.

Miscellaneous

Send Mail

Not fully implemented. Sending of multiple addresses and using the foundset to collate addresses is not yet implemented.

Navigation

Enter Preview Mode

Not fully implemented. See `Printing Implementation Details` section for more details.

Go To Layout

Not fully implemented. The 'with animation' form is not yet implemented.

Go To Record Request Page Number

Not fully implemented. The 'with dialog' form is not yet implemented.

Go To Related Record

Not fully implemented. The 'with animation' form is not yet implemented.

Go To Related Record In New Window

Not fully implemented. Only 'document' and 'card' styles are supported. Only the 'name', 'x', 'y', 'width' and 'height' new window options are honoured.

Print

Print

Not fully implemented. The 'with dialog' form is not yet implemented. See the `Printing Implementation Details` section for further details.

Records

Delete All Records

Not fully implemented. The 'with dialog' form is not yet implemented.

Duplicate Record Request

Not fully implemented. Duplicating a record in the context of a non master-detail portal is yet to be implemented.

Revert Record Request

Not fully implemented. The 'with dialog' form is not yet implemented.

Windows

Adjust Window

Not fully implemented. The Restore option is currently unimplemented.

New Window

Not fully implemented. Only 'document' and 'card' styles are supported. Only the 'name', 'x', 'y', 'width', 'height' and 'dim parent window' new window options are honoured.

Not Yet Implemented

Control

- Set Error Logging

Editing

- Copy
- Cut
- Paste
- Perform Find Replace

- Undo Redo

Fields

- Insert File Into Variable
- Insert From Index
- Insert From Last Visited
- Relookup Field Contents

Files

- Close Data File
- Convert File
- Create Data File
- Delete File
- Get Data File Position Into Field
- Get Data File Position Into Variable
- Get File Exists Into Field
- Get File Exists Into Variable
- Get File Size Into Current Field
- Get File Size Into Field
- Get File Size Into Variable
- Open Data File Into Field
- Open Data File Into Variable
- Open File With Dialog
- Read From Data File Into Field
- Read From Data File Into Variable
- Rename File
- Set Data File Position
- Write To Data File From Field
- Write To Data File From Variable

Miscellaneous

- AVPlayer Play Field
- AVPlayer Play Object
- AVPlayer Play URL
- AVPlayer Set Options
- AVPlayer Set Playback State
- Clear Region Monitor Script
- Configure Region Monitor Script Geolocation
- Configure Region Monitor Script Ibeacon

- Get Folder Path
- Install Menu Set
- Perform Apple Script
- Send DDEExecute File
- Send DDEExecute Topic
- Send Event With File
- Send Event With Message
- Set Layout Object Animation

Records

- Copy All Records Requests
- Copy Record Request
- Save Records As Excel
- Save Records As Snapshot Link
- Truncate Table

Spelling

- Check Current Field Selection
- Check Found Set
- Check Record
- Check Selection
- Correct Word
- Edit User Dictionary
- Select Dictionaries
- Spelling Options

Windows

- Arrange All Windows
- Enable Touch Keyboard
- Scroll Window
- Set Zoom Level
- Show Hide Menubar

Not Supported

Fields

- Insert Object
- Update Link

Files

- Close ODBCData Source
- New File
- Open ODBCData Source
- Recover File
- Save ACopy As
- Save ACopy As To File
- Save ACopy As XML

Menus

- Open Edit Saved Finds
- Open Favorites
- Open File Options
- Open Find Replace
- Open Help
- Open Hosts
- Open Manage Containers
- Open Manage Data Sources
- Open Manage Database
- Open Manage Layouts
- Open Manage Themes
- Open Manage Value Lists
- Open Preferences
- Open Script Workspace
- Open Sharing
- Open Upload To Host

Miscellaneous

- Execute SQL
- Install Plug In File From Field

Windows

- Set Allowed Orientations

Extended Script Steps

Licensing controls

The LCFM Native plugin provides script steps to manage your app's LCFM Native license.

Named User Licenses

Apps with a named user license require a user name to be registered. Use the `Register Named User` script step to register a user name f the license. If, after a period of grace, a user name has not been registered for the license, apps with a named user license will cease to functic

```
Register Named User [ Name: <name> ]
```

Any currently registered named user can be fetched using the `Get Registered Named User` function. This will be empty if no user name has been registered or if the app's license is not a named user license:

```
GetRegisteredNamedUser()
```

Functions

GetRegisteredNamedUser

Purpose

Get named user registered for license

Format

```
GetRegisteredNamedUser
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the name of the user currently registered for the license. If no user has been registered or the license is not a named user license, empty will be returned.

Example 1

Put the currently registered named user into the variable `$NamedUser`

```
Set Variable [ $NamedUser ; Value: GetRegisteredNamedUser() ]
```

Related topics

Register Named User script step

Script Steps

Register Named User

Purpose

Register a named user

Format

```
Register Named User [ Name: <value> ]
```

Options

- The name of the user to be registered

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Register a named user for the license.

Apps with a named user license require a user name to be registered. Use the `Register Named User` script step to register a user name f the license. If, after a period of grace, a user name has not been registered for the license, apps with a named user license will cease to functio

If the register operation is successful, `Get(LastError)` returns 0. If the named user passed is empty, then error code 5 (invalid command) i returned. If the app's license is not a named user license, then error code 3 (unavailable command) is returned.

Example 1

Register a named user with the content of field `UserName`

```
Register Named User [ Name:  GetField(UserName) ]
```

Related topics

Get Registered Named User function

Snapshot

The LCFM Native plugin provides script steps allowing apps to take snapshots of windows. The snapshots taken will be compressed PNGs an can be entered into a variable or field.

Script Steps

Take Snapshot Of Current Window Into Variable

Purpose

Take a snapshot of the current window

Format

```
Take Snapshot Of Current Window Into Variable
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Where the script step runs	Supported

Originated in

LCFM Native

Description

Takes a snapshot of the current window and enters it into a variable or field as a compressed PNG. You must specify a target variable, specify target field on the current layout, click in a field, or use the `Go to Field` script step before performing this script step. If Target is not specified, the data is placed in the active field. Otherwise, this script step returns an error code that can be captured with the `Get(LastError)` function. If specified, the target field must be a container field. Otherwise, this script step returns an error code that can be captured with the `Get(LastError)` function.

Example 1

Enter a snapshot of the current window into the variable `$Snapshot`

```
Take Snapshot Of Current Window [ Select ; Target: $Snapshot ]
```

Related topics

Take Snapshot Of Window script step

Take Snapshot Of Window Into Variable

Purpose

Format

```
Take Snapshot Of Window Into Variable
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native

Description

Sync controls

The LCFM Native plugin provides the ability to control the application's sync settings within FileMaker solution scripts, as well as the ability to actually trigger syncs manually.

If automatic sync is enabled, the application will sync with the server after every local commit, and also periodically when idle (at a frequency governed by the automatic sync interval). If automatic sync is disabled, sync can only be triggered explicitly through the `Sync` or `Request Sync` script steps.

Functions

GetAutomaticFullSyncFrequency

Purpose

Get the frequency of automatic full syncs

Format

```
GetAutomaticFullSyncFrequency
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the number of normal sync intervals before a full sync. If the app was compiled without sync support, then `0` is returned.

Related topics

Set Automatic Sync Interval script step

GetAutomaticFullSyncOnLocalCommit

Purpose

Determine whether automatic sync is full or not after local commit

Format

```
GetAutomaticFullSyncOnLocalCommit
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the value of the 'full sync on local commit' option, which determines whether deletion checks are performed when a sync is triggered b a local commit. If the app was compiled without sync support, then `False` is returned.

Related topics

Set Automatic Full Sync On Local Commit script step

GetAutomaticSyncEnabled

Purpose

Determine whether automatic sync is enabled or not.

Format

```
GetAutomaticSyncEnabled
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Determines whether the automatic syncing mechanism is enabled or not. If the app was compiled without sync support, then `False` is returned.

Example 1

Ensure sync happens whether automatic or not

```
If [ not GetAutomaticSyncEnabled() ]
        Request Sync
End If
```

Related topics

Set Automatic Sync Enabled script step

GetAutomaticSyncInterval

Purpose

Get the interval between automatic syncs

Format

```
GetAutomaticSyncInterval
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the interval between checks for updates on the server, when there have been no local changes. The interval is in seconds. An interval of 0 means sync will only happen when a local commit needs to be pushed. If the app was compiled without sync support, then `0` is returned.

Related topics

Set Automatic Sync Interval script step

GetLastSyncTimeUTCMilliseconds

Purpose

Get the time of the last sync

Format

```
GetLastSyncTimeUTCMilliseconds
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the time at which the last sync occurred in milliseconds since 1/1/0001 - i.e. it is comparable with `Get(CurrentTimeUTCMilliseconds).` If the app was compiled without sync support, then `0` is returned.

Related topics

Set Automatic Full Sync On Local Commit script step

GetTableSyncConstraint

Purpose

Get the sync constraint applied to a table

Format

```
GetTableSyncConstraint
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Returns the sync constraint that is applied to a table, previously set using the `Set Table Sync Constraint` script step. If the app was compiled without sync support, then empty is returned. type: calc label: Table datatype: Text showinline: true description: | The name of the tab

Related topics

Set Table Sync Constraint script step

GetTableSyncMode

Purpose

Get the sync mode of a table

Format

```
GetTableSyncMode
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Script Steps

Cancel Sync

Purpose

Cancel a pending sync request.

Format

```
Cancel Sync
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Cancels any pending sync request that might be present. When automatic syncing is enabled, `Cancel Sync` cancels the next sync whether be scheduled according to the automatic sync interval or after a local commit. If automatic sync is disabled, this cancels a pending sync requested using the `Request Sync` script step. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Prevent a local commit from being synced automatically

```
New Record/Request
Commit Records/Requests [ With dialog: Off ]
Cancel Sync
```

Example 2

Cancel an explicitly requested sync

```
Request Sync
Cancel Sync
```

Related topics

Request Sync script step

Delete All Local Records

Purpose

Delete all local records

Format

```
Delete All Local Records
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Planned but not yet implemented.

Delete Local Record

Purpose

Delete a local record

Format

```
Delete Local Record
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Planned but not yet implemented.

Disable Automatic Sync

Purpose

Disable automatic sync.

Format

```
Disable Automatic Sync
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Disables the automatic syncing mechanism. When automatic sync is disabled, no automatic syncing occurs at all - even if local changes are made. Sync is only triggered by the `Request Sync` and `Sync` script steps. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Prevent a local commit from being synced automatically

```
Disable Automatic Sync
New Record/Request
Commit Records/Requests [ With dialog: Off ]
```

Related topics

Enable Automatic Sync script step GetAutomaticSyncIsEnabled function

Enable Automatic Sync

Purpose

Enable automatic sync.

Format

```
Enable Automatic Sync
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Enables the automatic syncing mechanism. When automatic sync is enabled, sync is triggered by local commits and also periodically according to the 'automatic sync interval' property. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Sync a local commit as soon as possible

```
Enable Automatic Sync
New Record/Request
Commit Records/Requests [ With dialog: Off ]
```

Related topics

Disable Automatic Sync script step GetAutomaticSyncIsEnabled function

Omit All Journaled Records

Purpose

Omit all journaled records

Format

```
Omit All Journaled Records
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Planned but not yet implemented.

Request Sync

Purpose

Request a sync.

Format

```
Request Sync
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Requests a sync which will be performed as soon as possible. Requested syncs can be cancelled using the `Cancel Sync` script step. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Trigger a sync with a particular local commit, but not all.

```
Disable Automatic Sync
New Record/Request
Commit Records/Requests [ With dialog: Off ]
Request Sync
```

Related topics

Cancel Sync script step

Resync All Tables

Purpose

Resync all tables

Format

```
Resync All Tables
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Planned but not yet implemented.

Resync Table

Purpose

Resync a table

Format

```
Resync Table
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes

Where the script step runs	Supported
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Set Automatic Full Sync Frequency

Purpose

Set the frequency of automatic full syncs

Format

```
Set Automatic Full Sync Frequency [ Frequency: <value> ]
```

Options

- The number of sync intervals before a full sync

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Sets the number of normal sync intervals needed before a full sync (i.e. one including deletion checks) should occur. The frequency is the number of sync intervals. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Sync every 5 minutes but only check for deletions once per hour

```
Set Automatic Sync Interval [ Interval: 300 ]
Set Automatic Full Sync Frequency [ Frequency: 12 ]
Enable Automatic Sync
```

Related topics

Enable Automatic Sync script step Set Automatic Sync Interval script step

Set Automatic Full Sync On Local Commit

Purpose

Toggle whether automatic sync is full or not after local commit

Format

```
Set Automatic Full Sync On Local Commit [ Enabled: <value> ]
```

Options

- Whether automatic sync on local commit is full or not

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

returns 65536 (Sync not supported).

Example 1

Sync created record to server and check for any remote deletions

```
Enable Automatic Sync
Set Automatic Full Sync On Local Commit [ Enabled: True ]
New Record/Request
Commit Records/Requests [ With dialog: Off ]
```

Related topics

Enable Automatic Sync script step

Set Automatic Sync Enabled

Purpose

Toggle whether automatic sync is enabled or not.

Format

```
Set Automatic Sync Enabled [ Enabled: <value> ]
```

Options

- Whether automatic sync is enabled or not

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Toggles whether the automatic syncing mechanism is enabled or not. When automatic sync is enabled, sync is triggered by local commits and also periodically according to the 'automatic sync interval' property. When automatic sync is disabled, no automatic syncing occurs at all - even if local changes are made. Sync is only triggered by the `Request Sync` and `Sync` script steps. If the app was compiled without sync suppor then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Make local commits without syncing

```
Set Automatic Sync Enabled [ Enabled: False ]
New Record/Request
Commit Records/Requests [ With dialog: Off ]
```

Related topics

GetAutomaticSyncEnabled function

Set Automatic Sync Interval

Purpose

Set the interval between automatic syncs

Format

```
Set Automatic Sync Interval [ Interval: <value> ]
```

Options

- The number of seconds between automatic syncs

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Sets the interval between checks for updates on the server, when there have been no local changes. The interval is in seconds. If the interval i 0 then sync will only happen when a local commit needs to be pushed. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported).

Example 1

Sync every minute

```
Set Automatic Sync Interval [ Interval: 60 ]
Enable Automatic Sync
```

Related topics

Enable Automatic Sync script step

Set Table Sync Constraint

Purpose

Set the sync constraint applied to a table

Format

```
Set Table Sync Constraint
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Set a constraint on a table which is applied to the SQL used to fetch remote records to consider when the table is synced. A sync constraint is an SQL expression which can be concatenated as an 'AND' with any other SQL 'WHERE' clause. It must only reference fields in the table concerned, and no variable interpolation is done. i.e. The script step must be re-executed with an updated constraint if, say, it depends on the current user, and the current user changes. If the app was compiled without sync support, then `Get(LastError)` returns 65536 (Sync not supported). type: calc label: Table datatype: Text showinline: true description: | The name of the table type: calc label: Constraint showinline: tru description: | The constraint to apply.

Example 1

Only sync user-relevant data in table 'Data' when user logs in

```
Loop
    ... wait for Login to be pressed ...

    ... authenticate and exit loop if successful ...
End Loop

Set Table Sync Constraint [ Table: "Data"; Constraint: "userId = " & $$USERID ]

Enable Automatic Sync
```

Related topics

GetTableSyncConstraint function

Set Table Sync Mode

Purpose

Set the sync mode of a table

Format

```
Set Table Sync Mode
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Show Journaled Only

Purpose

Show only journaled records

Format

```
Show Journaled Only
```

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Planned but not yet implemented.

Sync

Purpose

Sync immediately

See also

How to control how a table syncs

Format

```
Sync [ Check For Deletions: <value> ; Timeout: <value> ]
```

Options

- Whether to check for remote deletions when syncing
- The number of seconds to attempt connection before timeout

Compatibility

Where the script step runs	Supported
LCFM Native - Android	Yes
LCFM Native - iOS	Yes
LCFM Native - macOS	Yes
FileMaker	No

Originated in

LCFM Native 1.5

Description

Runs a sync immediately, with optional deletion checks with the given timeout.

The timeout is the time to wait to try and contact the server and start a sync operation, not a timeout for the sync operation as a whole. If a syn does not start within the timeout period then the previous state of a sync request is unaffected - i.e. if 'Sync' was called when there was a pending sync request, and no sync occurs, the sync request will still be present; if 'sync' was called when there was no pending sync request, and no sync occurs, then there will still be no pending sync request.

In order to sync, there must be no locked records in any window. If this is not the case then the operation will have no effect and `Get(LastError)` returns 301 (record locked).

If the sync operation succeeds, `Get(LastError)` returns 0. If the connction times out, it returns 1629 (Connection timed out). If the app was compiled without sync support, then it returns 65536 (Sync not supported).

Otherwise, `Get(LastError)` returns one of the following: 1631: Connection failed 212: Invalid user account and/or password 802: Unable tc open file 65537: Error occurred during sync In this case GetLastExternalErrorDetail() returns a more specific error message.

If an error is generated during sync after a successful login (indicated by error 65537) then no modifications are made to local records, althoug some changes may have been synced to the remote server.

Example 1

Sync and check for deletions immediately, waiting at most 5 seconds to make a connection to FileMaker Server.

```
Sync [ Check For Deletions: True; Timeout: 5 ]
```

Related topics

Set Table Sync Constraint script step