Code-X User Manual

Demo Users Please Note

If you have not yet purchased a license for Code-X, you will not be able to integrate Code-X into your own solutions as the demo files are locked. This documentation is intended for licensed users, but is provided here to give new users an idea how easy it is to integrate Code-X into your own solutions.

Introduction

Welcome to the world of Code-X!

Protect your FileMaker solutions from piracy with Code-X. Use Code-X to generate custom registration codes for your FileMaker solutions and pass the codes to your paying customers so that they can unlock your trail solution.

Restrict your trial solutions so that users have an incentive to purchase a registration from you. Add 30-day trial periods, limit record creation, restrict features, it's entirely up to you. Please see the '**Feature Restriction**' example files.

With Code-X you have the ability to issue different levels of licensing. For example, you could have a Standard License and a Full License allowing different levels of functionality within your solution. You could issue a registration code that will unlock your solution for full access, or for standard access.

Use Code-X to restrict the number of users that can access your solution. Issue license upgrades that allow additional users to access your solution. Initially a customer may purchase a 5-user license from you – 3 months down the track they may purchase an upgrade for 10 users.

Use our "Code Manager" FileMaker database solution to generate and keep track of registrations. It's free, it's unlocked and it is included in the download. *(Please note the demo "Code Manager" is locked).*

Code-X gives you peace of mind – Code-X keeps your solution safe and protected.

Contents

Purchasing Code-X

If you haven't yet purchased Code-X, you will need to do so before you can implement Code-X within your own solutions.

To purchase Code-X: http://hi-voltage.com.au/products/filemaker-developer-tools/code-x/

Installing Custom Functions

Code-X uses custom functions that you will need to install in your solution.

To install the custom functions, you will need FileMaker Pro Advanced. If you do not have Advanced, for a fee Hi-Voltage can install the custom functions on your behalf.

Step 1: Copy Custom Functions from Examples.fmp12

A) Open the Examples.fmp12 file with FileMaker Pro Advanced

**** Please note:*** *The Examples.fmp12 file within the Code-X Demo is permanently locked and cannot be used to access the custom function. You must purchase Code-X and use the unlocked Examples.fmp12 file.*

B) Open the Custom Functions window

C) Select all Custom Functions and copy them to your clipboard

Step 2: Paste Custom Functions into your solution

A) Open your solution in FileMaker Pro Advanced.

B) Open the Custom Functions window

C) Paste the contents of your clipboard into the window

Congratulations, you have now installed the custom functions.

Installing Scripts

In order to use Code-X within your solution, you need to install the required Code-X scripts. To install the scripts, please follow the steps bellow:

Step 1: Copy required Scripts from Examples.fmp12

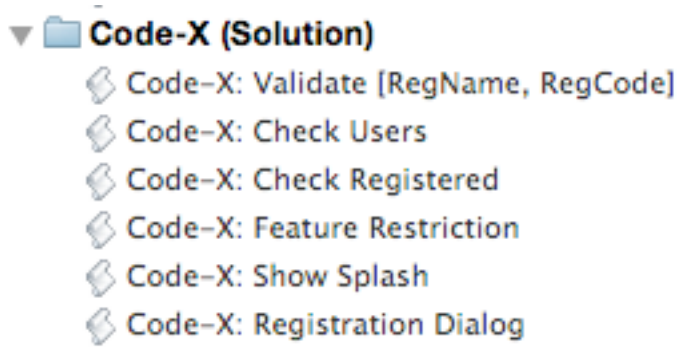
A) Make sure you have first installed the Custom Function, if not please see the section Installing Custom Function above.

B) Open the Examples.fmp12 file with FileMaker

**** Please note:*** *The Examples.fmp12 file within the Code-X Demo is permanently locked and cannot be used to access the custom function. You must purchase Code-X and use the unlocked Examples.fmp12 file.*

B) Open 'Scripts Workplace' in FM14+ ('Manage Scripts' in earlier versions)

C) Select the folder 'Code-X (Solution)' and copy all the scripts within this folder to your clipboard:



Step 2: Paste Scripts into your solution

A) Open your solution in FileMaker.

B) Open 'Scripts Workplace' in FM14+ ('Manage Scripts' in earlier versions)

C) Paste the contents of your clipboard into the window

Congratulations, you have now installed your scripts.

Implementing Code-X

After you have installed all the components required by Code-X (i.e. Custom Function and Scripts), you are ready to implement Code-X within your own solution.

Step 1: Create Solution entry within Code Manager

A) Open the 'Code Manager' file that came with Code-X and navigate to the 'Solutions' tab.

B) Click on the New button to create a new Solution entry and a unique Solution Key is generated for you automatically. DO NOT MODIFY THIS KEY.

C) Copy the entire Solution Key to your clipboard.

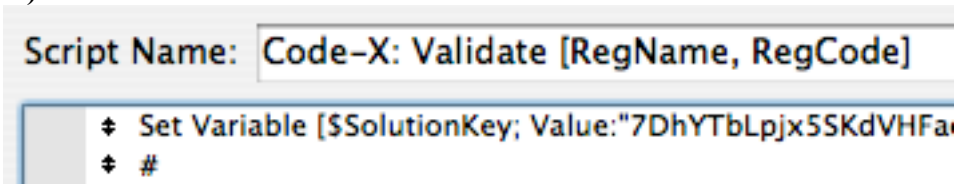
D) Enter the name of your solution within the Solution Name field.

Step 2: Paste the Solution Key into your solution

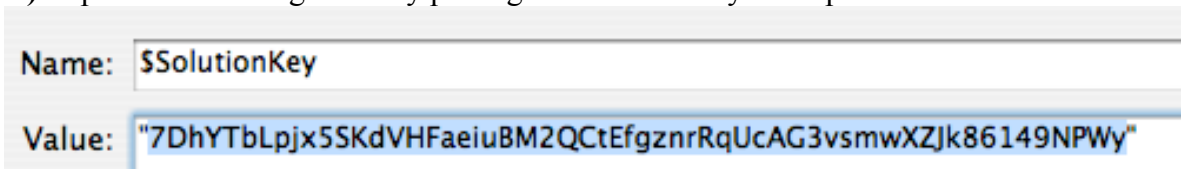
A) Open your solution

B) Open the script **Code-X: Validate**

C) Edit the first line of code 'Set Variable':



D) Replace the existing value by pasting the contents of your clipboard.



E) Save your script.

Step 3: Create RegName and RegCode fields

A) Choose a table within your solution where you would like to create the RegName and RegCode fields, i.e. within a Globals table. If you don't have anywhere suitable, you can create a new table.

B) Create two global text fields, one called 'gRegCode' and the other 'gRegName'.

Step 4: Create Registered and UserCount fields

A) Choose a table within your solution where you would like to create the 'Registered' field and UserCount field, i.e. within a Preferences table. If you don't have anywhere suitable, you can create a new table. Make sure there's a record within the table.

B) Create a number field called 'Registered' within the table.

NOTE: This field will be used to flag your solution as Registered or Unregistered. If this field is blank (false), it means your solution is unregistered. If this field is set to 1 (true), it means your solution is Registered. Your startup script will check if the solution is registered or not by referencing this field. You can then perform different scripts depending whether or not the solution is registered.

C) If you want to restrict the number of users, create a new number field called 'UserCount' within the table. You can skip this step if you're not restricting the user count.

NOTE: This field will store the number of users your license allows so that you can restrict how many users can access the your solution.

Step 5: Edit the Code-X: Registration Dialog script

A) Open the script 'Code-X: Registration Dialog' within your solution

-- MODIFICATION 1 --

B) Edit the 'Show Custom Dialog' script step and choose the tab 'Input Fields'

C) Click on the first 'Specify' button, the one next to 'Show input field #1' and choose the new global field you created earlier called 'gRegName'.

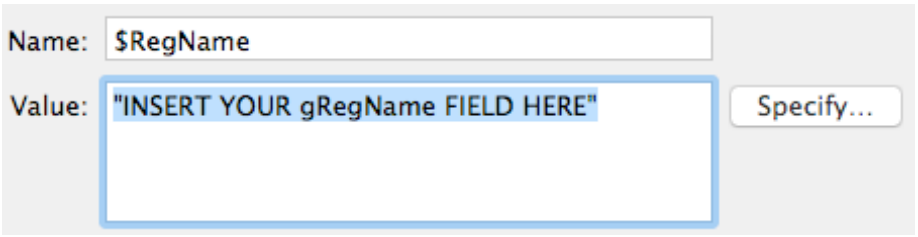
D) Click on the second 'Specify' button, the one next to 'Show input field #2' and choose the new global field you created earlier called 'gRegCode'.

E) Click the 'OK' button to save your changes

-- MODIFICATION 2 --

F) Edit the 'Set Variable' script step

G) Replace the contents in the 'Value' section and specify your gRegName field – use the Specify button to help you locate the field.

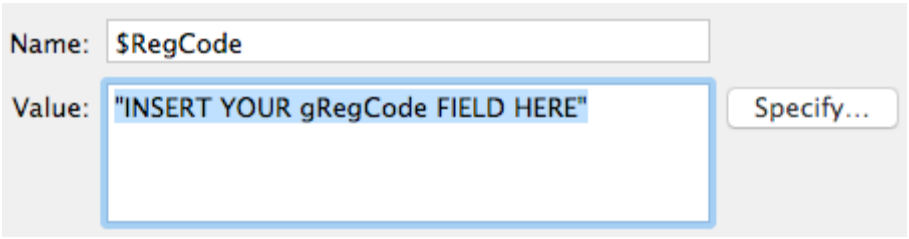


H) Click the 'OK' button to save your changes

-- MODIFICATION 3 --

I) Edit the 'Set Variable' script step

J) Replace the contents in the 'Value' section and specify your gRegCode field – use the Specify button to help you locate the field.



K) Click the 'OK' button to save your changes

-- MODIFICATION 4 --

L) Edit the 'Go to Layout' script step and specify the layout of the table where you created your 'Registered' and 'UserCount' fields earlier.

-- MODIFICATION 5 --

M) Double click the ‘Set Field’ script step and specify the ‘Registered’ field you created earlier.

-- MODIFICATION 6 --

N) Double click the ‘Set Field’ script step and specify the ‘UserCount’ field you created earlier.

NOTE: If you are not restricting the number of users, then you can delete this script step.

-- MODIFICATION 7 --

O) Edit the ‘Open URL’ script step and specify the URL address of your purchase page so your customers can buy a license for your solution.

-- DONE --

P) Save your script to save your changes

Congratulations you have implemented Code-X in your solution

Restricting features

If you want to restrict a feature, i.e. a button so it is only available to registered users, you can use the script ‘Code-X: Feature Restriction’.

Simply edit the script of the button that you want to restrict and add a Perform Script step as the very first line of code and call the ‘Code-X: Feature Restriction’ script.

Example (your button’s script):

```
Perform Script [“Code-X: Feature Restriction”]
# rest of your script here
```

Now when a user clicks on your button that you have restricted, it will check if your solution is registered or not.

If it is not registered, then it will show the user a message informing them the feature is only available in the registered version, and then it will halt the main script from running.

If it is registered, then the script will continue to run normally.

See ‘Script Documentation’ at the end of this document for more information.

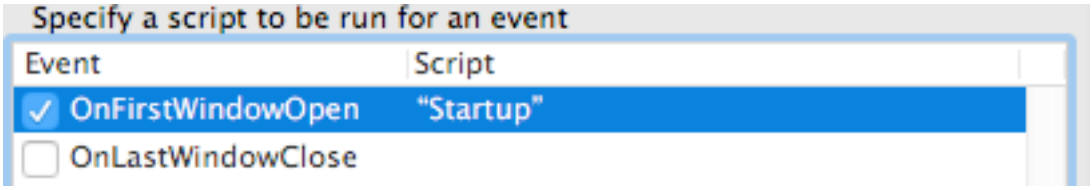
Creating a Start Up script

For some features you may need to create a startup script if you don't already have one. We'll advise when you need to have a start up script throughout this document.

A) Create a new script called "Startup" – leave the script empty for now with no script steps. Make sure you save it.

B) From the 'File' menu, choose 'File Options', then click on the 'Script Triggers' tab.

C) Double click on the "OnFirstWindowOpen" event, and specify your 'Startup' script.



D) Click 'OK' button to save your changes.

Congratulations you have created your startup script.

Show splash at startup

If you want to show the user a splash page at start up when the solution is not registered (i.e. to prompt them to register), you can do this by modifying your startup script.

Step 1: Modify Startup script

A) Edit your Startup script. If you don't already have one, see section 'Creating a Startup script'.

B) If you already have an existing startup script, you will need to determine where exactly you need to modify your code and add a new Perform Script step. We cannot help you here, as we don't know what your code looks like.

Basically, you need to determine where the startup script should check if your solution is registered, then present the user a splash page if it's not registered or go to the main page if it is registered.

C) Add a Perform Script step, and call the 'Code-X: Show Splash' script. Save your script.

Step 2: Modify Code-X: Show Splash script

A) Open the 'Code-X: Show Splash' script and apply the following changes:

-- MODIFICATION 1 –

B) Edit the 'Go to Layout' script step and specify the splash layout. If you don't have a splash layout, create one first.

-- MODIFICATION 2 --
C) Edit the 'Go to Layout' script step and specify the main layout. This will be the main layout where users go when using your solution, i.e. a dashboard, main menu etc.

Congratulations you have added a splash page

Restrict the number of users

You can restrict the number of users that are allowed to access your solution at any one time. You would normally do this at start up.

Step 1: Modify Startup script

A) Edit your Startup script. If you don't already have one, see section 'Creating a Startup script'.

B) If you already have an existing startup script, you will need to determine where exactly you need to modify your code and add a new Perform Script step. We cannot help you here, as we don't know what your code looks like.

Basically, you need to determine where the startup script should check how many users are currently connected.

C) Add a Perform Script step, and call the 'Code-X: Check Users' script. Save your script.

Step 2: Modify Code-X: Check Users script

A) Open the 'Code-X: Check Users' script and apply the following changes:

-- MODIFICATION 1 –

B) Edit the 'Go to Layout' script step and specify the layout of the table where you created the UserCount field.

Congratulations you have restricted the number of users.

Script Documentation

Code-X: Validate [RegName, RegCode]

Parameters:

- RegName (the Registration Name)
- RegCode (the Registration Code)

Return Result:

- Boolean: True = validation passed, False = validation failed

Explanation:
You call this script in order to validate a Registration Name against a Registration Code to see if they match.

If validation is successful, the script returns a true value. If validation fails, the script returns a false value.

Example:

```
Perform Script [“Code-X: Validate [RegName, RegCode]”; Name|Code ]
If [ Get(ScriptResult) = True ]
    Show Custom Dialog [“Success”]
Else
    Show Custom Dialog [“Fail”]
End if
```

The example above validates the Registration Name and Registration Code, and alerts the user if the validation was successful or not.

Code-X: Check Users

Parameters:
- none

Return Result:
- none

Explanation:
You call this script in order to check if your solution exceeds the number of allowed users. If it does then it shows the user a message and quits out of FileMaker. You would normally call this script from your startup file, so that when your solution starts up, it checks the number of users and if it exceeds the maximum allowed.

Example:
```
Perform Script [“Code-X: Check Users”]
```

The above example simply calls the Check Users script. If the number of users has been exceeded, then FileMaker quits.

Code-X: Check Registered

Parameters:

- none

Return Result:

- Boolean (True = solution is registered, False = solution not registered)

Explanation:
You call this script in order check if your solution is registered or not. True is returned if your solution is registered, and False is returned if your solution is not registered.

Example:

```
Perform Script [“Code-X: Check Registered”]
If [ Get(ScriptResult) = True ]
    Show Custom Dialog [“Registered”]
Else
    Show Custom Dialog [“Not Registered”]
End if
```

The example above checks if the solution is registered or not. If it is registered it shows the message “Registered”, otherwise if it is not registered it shows the message “Not Registered”.

You could use a variation of this script to check at start up if the solution is registered or not and show a splash page if it’s not registered asking your users to register.

Code-X: Feature Restriction

Parameters:
- none

Return Result:
- none

Explanation:
You call this script in order to restrict a feature if the solution is not registered, i.e. another script.

Lets say you want to restrict a button. The buttons script should be modified and the first line of code should call the Code-X: Feature Restriction script. The rest of your code goes under this first line of code.

If the solution is unregistered then a dialog message is shown informing the user that the feature is restricted, and then the main script is halted. If the solution is registered the main script continues to run normally.

Example:
```
Perform Script [“Code-X: Feature Restriction”]
# rest of your script here
```

The example above calls the Code-X Feature Restriction script to check if the solution is registered or not. If the solution is not registered then the main script will be halted. If the solution is registered then the main script will continue to run normally.

Code-X: Show Splash

Parameters:
- none

Return Result:
- none

Explanation:
You call this script in order to show a splash page if the solution is not registered. Normally you would call this script from your start up script so that at startup the user is shown a splash screen if not registered, prompting them to register.

Example:
```
Perform Script [“Code-X: Show Splash”]
```

The example above simply calls the Show Splash script. If the solution is unregistered, it takes the user to the splash page, if the solution is registered it takes the user to the main layout.

Code-X: Registration Dialog

Parameters:
- none

Return Result:
- none

Explanation:
You call this script in order to allow demo users to register your solution.

Example:
```
Perform Script [“Code-X: Registration Dialog”]
```

The example above simply calls the Registration Dialog script, which shows the user a Dialog allowing them to enter their Registration Name and Registration Code, and performs a validation. If the validation is successful, the then flags the solution as registered.

Custom Function Documentation

GetScriptParameter(Number)

Parameters:

- Number (the nth script parameter to check)

Return Result:

- Text (the nth script parameter as specified by Number)

Explanation:
You call this custom function in order obtain the nth script parameter. This custom function allows us to pass multiple script parameters and extract them by number reference in our scripts.

It is similar to the native FileMaker function Get(ScriptParameter), except this function only caters for one parameter. Our custom function works with multiple parameters.

Example:

```
Passing multiple parameters to script:
Perform Script [“Code-X: Validate”; Parameter: “Name|Code”]


Extracting parameters in script:
Set Variable [$RegCode; Value: GetScriptParameter(1)]
Set Variable [$RegName; Value: GetScriptParameter(2)]
```

The first example above passes two script parameters within a script, the parameters must be separated by the pipe command “|”.

The second example extracts each script parameters by referencing to it by position and stores the value in a variable.

GetUserCount(RegCode)

Parameters:

- RegCode (the registration code you're checking for user count)

Return Result:

- Number (the number of users the registration code is licensed for)

Explanation:
You call this custom function in order obtain the number of users the Registration Code is licensed for.

Example:

```
Set Variable [$UserCount; Value: GetUserCount($RegCode)]
```

The example above sets a variable called $UserCount with the number of users the registration code contained in the $RegCode variable.

