

.com Solutions Inc.



# Code Conversion Workbench Manual

# Code Conversion Workbench Manual

## 1 Overview

1.1	Overview	4
-----	----------	---

## 2 GUI

2.1	Using the Code Conversion Workbench	6
-----	-------------------------------------	---

## 3 Detailed Code Conversion Info

3.1	Using the VFP Code Conversion Workbench	19
-----	---	----

3.2	Using the Microsoft Access Code Conversion Workbench - Access to FmPro Conversions	24
-----	--	----

3.3	Using the FmPro Code Conversion Workbench - FmPro to Access Conversions	28
-----	---	----

3.4	Using the FmPro Code Conversion Workbench - FmPro to Servoy Conversions	32
-----	---	----

3.5	Using the FmPro Code Conversion Workbench - FmPro to PHP Conversions	36
-----	--	----

3.6	Using the FmPro Code Conversion Workbench - FmPro to LiveCode Conversions	40
-----	---	----

3.7	Using the VFP Code Conversion Workbench - Visual FoxPro to .NET Conversions	44
-----	---	----

3.8	Using Google Gemini Models	48
-----	----------------------------	----

3.9	Installing Ollama for Local LLM Usage	49
-----	---------------------------------------	----

## 4 Troubleshooting

4.1	Troubleshooting - Code Conversion Workbench	56
-----	---	----

# Overview

## Overview

---

The Code Conversion Workbench feature within the [AI Accelerated version of FmPro Migrator Platinum Edition](#) is used to convert code from FileMaker Pro, FoxPro 2.6, Microsoft Access and Visual FoxPro. The Code Conversion Workbench converts code into over 50 different programming languages by using multiple machine learning models. As the code is converted, the files are written into the output directory, and can also be copied via the clipboard and pasted directly into FileMaker Pro or other development environments.

Revision 3

FmPro Migrator 11.13

5/11/2024

[Revision Notes: Added network troubleshooting info, added Google Gemini info, added info showing how to run machine learning models locally with Ollama using FmPro Migrator Custom Dev Edition.]

# GUI

## Using the Code Conversion Workbench

---

The Code Conversion Workbench is included with the AI Accelerated version of FmPro Migrator Platinum Edition.

The Code Conversion Workbench consists of a single application window used to manage the conversion of hundreds or thousands of scripts into over 50 programming languages. As scripts are completed, they can be checked off in the grid of script on the left side of the window.

The source of scripts used by the Code Conversion Workbench is the MigrationProcess.db3 SQLite database file which is created and used by FmPro Migrator. Before performing code conversion, the scripts need to already be imported into the Scripts tab of the Migration Process window of FmPro Migrator.

Each source database is used to prefix the name of the code conversion, so if Visual FoxPro is the source database type, the title across the window will be: VFP Code Conversion Workbench.

Most screenshots in this manual are from macOS, but the tool works exactly the same way on Windows.

## Code Conversion Workbench Window

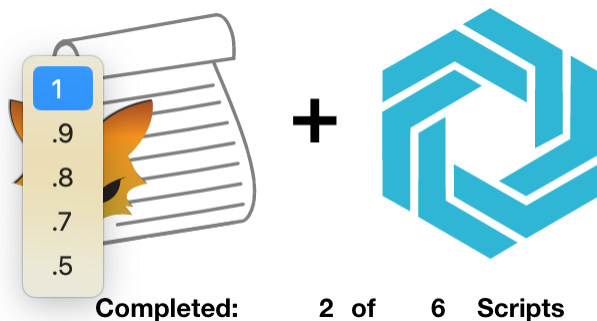
The screenshot shows the VFP Code Conversion Workbench interface. On the left, a table lists 6 scripts with their status, ID, size, and name. Callout 1 points to the fox logo, and callout 2 points to the 'Load Data' button. The table data is as follows:

Status	ID	Size	Script Name
Not Started	1	2441	pgraph.prg
Not Started	2	4760	cgraph.prg
Completed	3	6788	main.prg
Not Started	4	14406	fdproc.prg
Not Started	5	20234	datepick.prg
Completed	6	1207	frmanimation

The right side of the window contains a 'Source Script (Editable)' area with VFP code (callout 14), a 'Command (Editable)' area for instructions (callout 15), and a 'Converted Script' area (callout 17). On the far right, a configuration panel includes dropdowns for 'Source' (Visual FoxPro, callout 4), 'Output Language' (C#, callout 5), 'Procedure/Function' (cmdok.Click 685-749, callout 6), 'Section' (1, callout 7), 'Vendor' (OpenAI, callout 8), 'API Key' (dcsi, callout 9), 'Model' (gpt-3.5-turbo, callout 10), 'Tokens Used Today' (713, callout 12), and 'Output Filename' (datepick\_cmdok\_Click.cs, callout 13). A 'Convert' button (callout 16) and a 'Refresh' button (callout 11) are also present. A 'Need Help?' button (callout 18) is at the bottom right.

Most features in this window include tooltips, described in this manual.

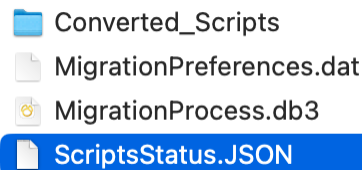
### 1) Window Scale Factor - Pop-up Menu



The Code Conversion Workbench window will auto-scale in size based upon the size of your display. You can change the window scaling by selecting a different value from this popup menu. The Window Scale Factor Menu tooltip will be displayed when you move the cursor over this

normally hidden popup menu.

## 2) Load Data Button



If there are scripts located within the FmPro Migrator MigrationProcess.db3 SQLite project file, the grid of scripts should be displayed automatically as soon as the window opens. If there weren't any scripts in the MigrationProcess.db3 file when the Code Conversion Workbench window was opened, the list of scripts would be empty. Once you import scripts using FmPro Migrator, click the Load Data button to populate this grid with the list of scripts.

The scripts displayed in this grid are read from the MigrationProcess.db3 file and the Status checkmarks/Completed/Not Started text are read from the ScriptsStatus.JSON file shown above.

**Note:** If the Code Conversion Workbench is running in Demo mode, only a list of demo scripts will be displayed, not the actual scripts within the MigrationProcess.db3 file.

## 3) Scripts Grid

Completed: 2 of 6 Scripts		Load Data		
Status	ID	Size	Script Name	
<input type="checkbox"/> Not Started	1	2441	pgraph.prg	
<input type="checkbox"/> Not Started	2	4760	cgraph.prg	
<input checked="" type="checkbox"/> Completed	3	6788	main.prg	
<input type="checkbox"/> Not Started	4	14406	fdproc.prg	
<input type="checkbox"/> Not Started	5	20234	datepick.prg	
<input checked="" type="checkbox"/> Completed	6	1207	frmanimation	

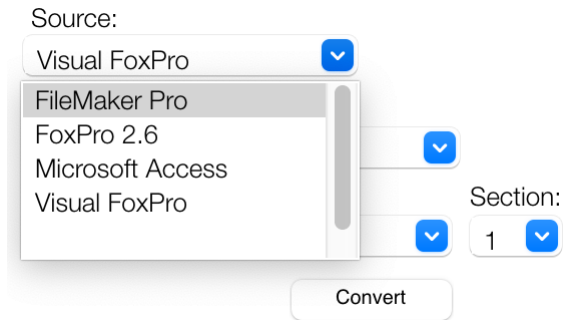
The scripts grid provides a way for the developer to select specific scripts, convert them and check them off as being completed by clicking the checkmark column. The Completed ? of ? labels above the grid give a running count of the conversion status of the scripts.

Each script has an ID for reference purposes and its size in bytes and name are listed in the grid. Long script names are viewable with the scrollbar at the bottom of the grid.



Clicking any script causes it to be read from the database and displayed in the Source Script field. Clicking the script a 2nd time (to set its check/unchecked status) does not reload the script - in order to keep the contents of the Converted Script field visible.

#### 4) Source Database Type Menu



Source database types include: FileMaker Pro, FoxPro 2.6, Microsoft Access and Visual FoxPro.

Selecting the Source Database changes the title across the top of the window and re-writes the contents of the Command window.


## 5) Output Language Menu

Source:  
Visual FoxPro

Output Language:  
C#

- ABAP
- Ada
- Assembly
- Awk
- Bash
- C
- C#
- C++
- CFML
- Classic Visual Basic
- COBOL
- D
- Dart
- Delphi/Object Pascal
- Emacs Lisp
- F#
- FileMaker Pro
- Fortran
- Go
- Groovy

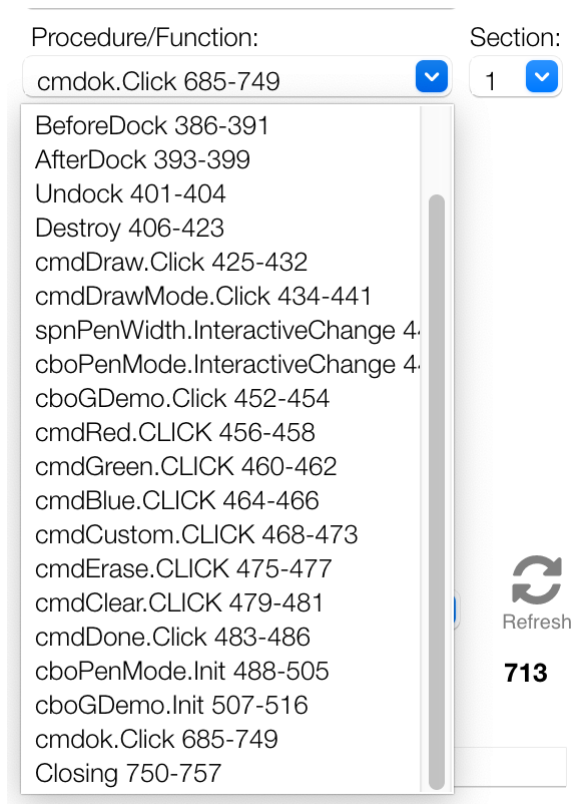
Section:  
1

  
Refresh

**713**

Selecting one of the over 50 output languages from the TIOBE index will rewrite the Command prompt for the selected language. Some of the languages include the specification of a framework and database name based upon the database selected on the main screen of FmPro Migrator as the output database. The text in the Command field can be manually updated to include any additional details required for the conversion.

## 6) Procedure/Function Menu



If the main script is too large for processing by a specific machine learning model, red text "Script Too Large" will be displayed above the Source script field. If this occurs, then it is possible to break of some scripts (like Visual FoxPro) automatically into individual procedures/functions individually.

Visual FoxPro scripts often contain multiple procedures/functions. It is generally better to send smaller pieces of text to the AI models for processing, so this menu enables selecting individual procedures/functions for sending to the AI model for processing.

## 7) Section Menu

Source Script (Editable): **Script Too Large** Size: 4,759

```
***** cgraph.prg
***** C:\DS\VFP_TESTS\SOLUTION\forms\graphics\cgraph.prg
* Plot graph (Polar).
* Parameter:
* 1) cEquation (For radius) in terms of X. TYPE = Character.
* 2) nFrom. Where to stop counting for X. TYPE = Numeric.
* 3) nTo. Where to start counting for X. TYPE = Numeric.
* 4) nStepInc. Step increment. TYPE = Numeric.
* 5) nEquColor. TYPE = Numeric.
* 6) IConnect. If the previous point is connected to the current point with a line.
TYPE = Logical.
* 7) nXCenter. Point on form where x = 0. TYPE = Numeric.
```

Command (Editable):

Translate this Visual FoxPro code into C# code

Source:

Visual FoxPro

Output Language:

C#

Procedure/Function:

Closing 1-132

Section:

1

Convert

Vendor:

OpenAI

1

2

3

4

5

Some individual procedure/functions could still be too large, including the example shown here. If the "Script Too Large" script is displayed for an individual procedure/function then the Section sub-menu will also be displayed. This enables the large script to be broken up into 5 more sections (with opening/closing procedure/function text added to each section).

By the way, just because the "Script Too Large" text is displayed doesn't always mean that a script won't be processed by the AI model. You can always try sending a large script for processing, and then see if you get an error from the model.

## 8) Vendor Menu

Vendor:

OpenAI

AWS

Azure


Google

OpenAI

Ollama



There are 5 AI vendors listed in the Vendor model at this time. OpenAI, Google and Ollama are the vendors supported at this time, as the others are intended for future enhancements.

## 9) API Key Type

API Key:    
Model:



By default, the Code Conversion Workbench will use the API key created by .com Solutions Inc., so "dcsi" is the default option. If you run out of AI tokens, you can generate and pay for your own API key at OpenAI.

## User API Key Field

API Key:    URL

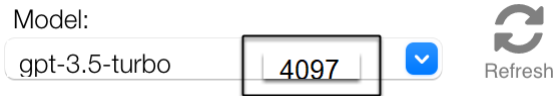
To enter your own API key, select the User menu item, a new API Key field will be shown. Clicking the URL button opens a web browser to the OpenAI website where you can create an account and generate your own API key. This key gets saved with the app preferences so you only have to enter it once.

## 10) AI Model Name

Model:    Refresh  
  
  
gpt-3.5-turbo  
gpt-3.5-turbo-0301  
gpt-3.5-turbo-0613  
gpt-3.5-turbo-16k  
gpt-3.5-turbo-16k-0613  
gpt-4  
gpt-4-0314  
gpt-4-0613  
text-davinci-001  
text-davinci-002  
text-davinci-003

gpt-3.5-turbo will be selected as the default model. gpt-4 is also available, and generally does a significantly better job when performing conversions. But sometimes one model or the other gets overloaded so you might want to switch to another one.

## Model Token Size Tooltip



Hovering the cursor over the model menu, displays the number of tokens accepted by the selected AI model.

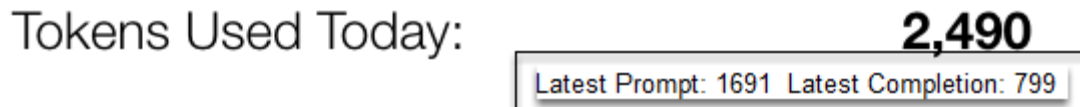
## 11) Model List Refresh Button



The list of available AI models is built into the app during development. You can refresh the list of models by clicking the Refresh button. The list of models displayed in the Model menu represents the models which are officially supported by the Code Conversion Workbench. At this time, there are 11 models in the list. Holding the Shift key when clicking the Refresh button displays all of the models available from the selected vendor. Selecting one of the unsupported models will generally result in an error message.

Updating the list of models with the refresh button doesn't permanently save these models to the Model menu when the app is closed.

## 12) Tokens Used Today Label & Usage Tooltip



The number to the right of the Tokens Used Today text label shows the number of tokens used over the last 24 hours. Hovering the cursor over this number will display the number of tokens used during the most recent script conversion. In this example, 1691 tokens were used for the prompt and 799 tokens were used for the completed script which was returned by the model.

## Tokens Available per Day Tooltip

Tokens Used Today:

2,490

500,000 Tokens Available per Day

Hovering the cursor over the Tokens Used Today label will display the number of tokens which can be used per day for the currently selected AI model. At the present time, 500,000 tokens per day are provided for gpt3.5-turbo and the rest of the models except for gpt4. Since it currently costs 20 times as much to use the gpt4 model, the number of tokens available per day is reduced to 25,000 when gpt4 is being used. These calculations will change over time as costs change.

## 13) Output Filename Field

Output Filename:

cgraph\_Closing.cs

The name of the output script is created and saved automatically into the Converted\_Scripts folder, but it can be changed manually before pressing the Convert button. In this example the filename consists of the name of the script followed by the name of the function and ending with the C# file extension.

## Output Filename Tooltip

Output Filename:

cgraph\_Closing.cs

cgraph\_Closing\_2023-08-24\_15\_7\_46.cs

The tooltip shows the actual filename written to disk. Duplicated scripts receive a timestamp in order to preserve the original script, as you can see with this example.

## 14) Source Script Field

**Source Script (Editable):** **Script Too Large** **Size:** 4,759

```
***** cgraph.prg
***** C:\DS\WFP_TESTS\SOLUTION\forms\graphics\cgraph.prg
* Plot graph (Polar).
* Parameter:
* 1) cEquation (For radius) in terms of X. TYPE = Character.
* 2) nFrom. Where to stop counting for X. TYPE = Numeric.
* 3) nTo. Where to start counting for X. TYPE = Numeric.
* 4) nStepInc. Step increment. TYPE = Numeric.
* 5) nEquColor. TYPE = Numeric.
* 6) lConnect. If the previous point is connected to the current point with a line.
TYPE = Logical.
* 7) nXCenter. Point on form where x = 0. TYPE = Numeric.
```

Each script is read from the MigrationProcess.db3 SQLite project file when it is clicked in the scripts list grid. The script is written into the Source Script field, which can be manually edited if needed.

The size of the script in characters is displayed above the top right of the field next to the Size label. If the size of the script is estimated to be too large for the number of tokens available for the selected model, the "Script Too Large" text will be displayed above the script.

## 15) Command Field

**Command (Editable):**

```
Translate this Visual FoxPro code into C# code
```

This field provides the prompt which will be sent to the AI model. The default command for this field is created automatically when selecting the Source database type or Output Language. It is also editable with additional descriptive commands such as the programming language framework or database type.

## 16) Convert Button

Convert

The Convert button sends the Source script, Command and system message/properties to the AI model for processing. The results are written into the Converted Script field.



## 17) Converted Script Field

### Converted Script:

Here is the translated code in C#:

```
```csharp
using System;
using System.Drawing;
using System.Windows.Forms;

public class CGraph
{
    public static void PlotGraph(string cEquation, double nFrom, double nTo, double
nStepInc, int nEquColor, bool lConnect, int nXCenter, int nYCenter, bool
lAddCoords, Form frmFormName, double nEquScale)
    {
        if (string.IsNullOrEmpty(cEquation) || frmFormName == null)
        {
            MessageBox.Show("This program requires multiple parameters.");
            return;
        }

        if (nEquScale == 0)
        {
            nEquScale = 1;
        }
    }
}
```

After clicking the Convert button, the generated script is put into the Converted Script field, and also written into the Converted\_Scripts folder within the project directory.

## 18) Clipboard Icon



Clicking the clipboard icon copies the contents of the Converted Script field onto the clipboard, ready for pasting into your IDE of choice.

When FileMaker Pro is selected as the Output Language the converted script text is converted into FileMaker script XML code and placed onto the clipboard in a format which can be directly pasted into the FileMaker Pro Script Workspace window.

# Detailed Code Conversion Info

## Using the VFP Code Conversion Workbench

---

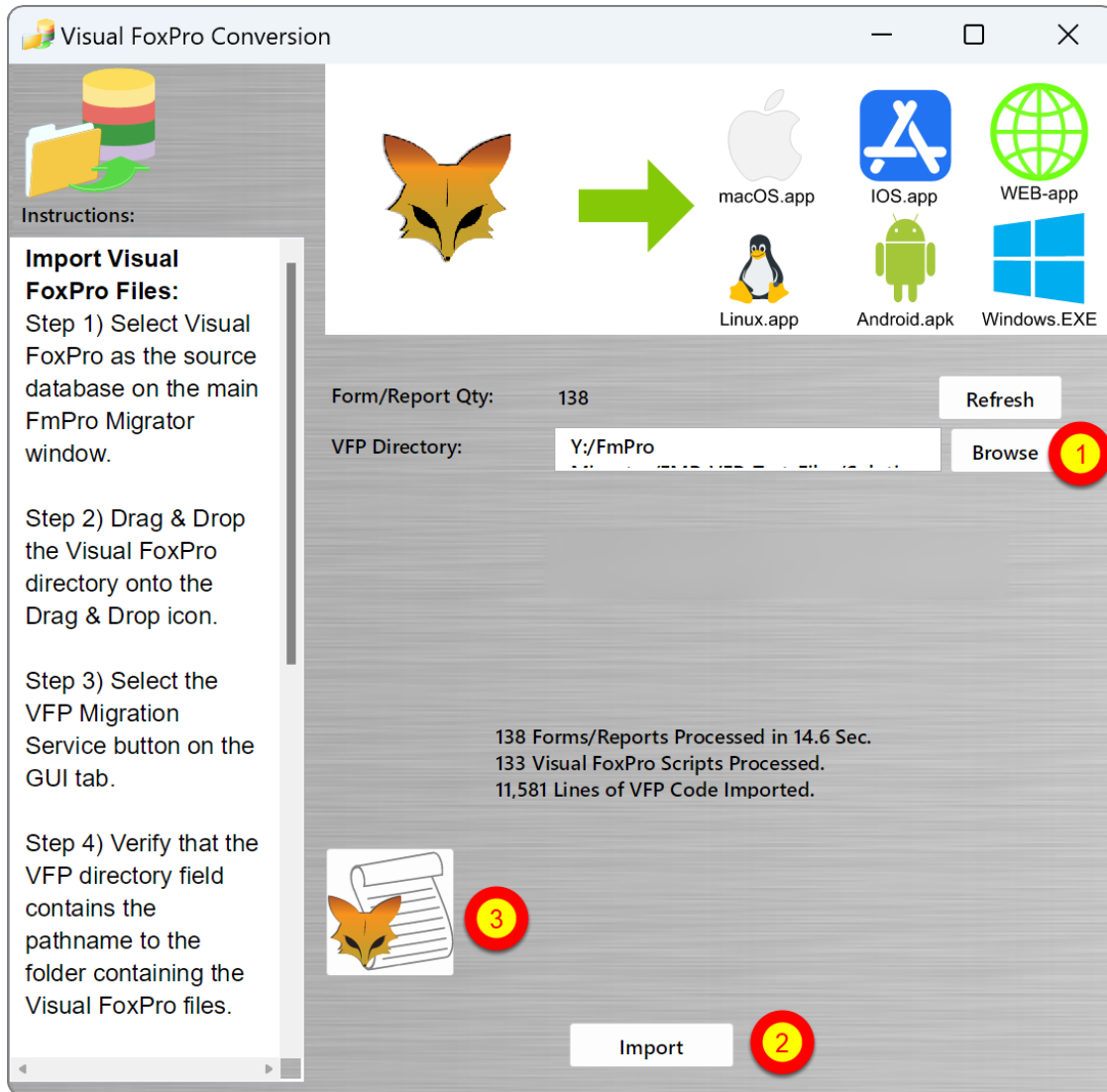
This chapter shows how to use the VFP Code Conversion Workbench. Before performing the code conversion with the VFP Code Conversion Workbench, you should have already imported the Visual FoxPro project into FmPro Migrator, including tables, relationships, value lists, forms/reports and scripts.

### Visual FoxPro Conversion Button - GUI Tab



Click the [Visual FoxPro Conversion](#) button on the GUI tab of the Migration Process window of FmPro Migrator.

## Visual FoxPro Conversion Window



Prior to reaching this step, you should have already selected the (1) selected the VFPEXport.DBF project folder and (2) clicked the Import button to import the VFP project into FmPro Migrator. You may have already started converting the VFP project into another development environment which would mean that you are ready (3) to click the Code Conversion Workbench button.

# VFP Code Conversion Workbench Window

Code Conversion Workbench

## VFP Code Conversion Workbench

Completed: 2 of 6 Scripts Load Data

Status	ID	Size	Script Name
<input type="checkbox"/> Not Started	1	2441	pgraph.prg
<input type="checkbox"/> Not Started	2	4760	cgraph.prg
<input checked="" type="checkbox"/> Completed	3	6788	main.prg
<input type="checkbox"/> Not Started	4	14406	fdproc.prg
<input type="checkbox"/> Not Started	5	20234	datepick.prg
<input checked="" type="checkbox"/> Completed	6	1207	frmanimation

**Source Script (Editable):** Script Too Large Size: 4,759

```
***** cgraph.prg
***** C:\DS\VFP_TESTS\SOLUTION\forms\graphics\cgraph.prg
* Plot graph (Polar).
* Parameters:
* 1) cEquation (For radius) in terms of X. TYPE = Character.
* 2) nFrom. Where to stop counting for X. TYPE = Numeric.
* 3) nTo. Where to start counting for X. TYPE = Numeric.
* 4) nStepInc. Step increment. TYPE = Numeric.
* 5) nEquColor. TYPE = Numeric.
* 6) lConnect. If the previous point is connected to the current point with a line.
TYPE = Logical.
* 7) nXCenter. Point on form where x = 0. TYPE = Numeric.
```

**Command (Editable):**  
Translate this Visual FoxPro code into C# code

**Converted Script:**  
Here is the translated code in C#:

```
''' csharp
using System;
using System.Drawing;
using System.Windows.Forms;

public class CGraph
{
    public static void PlotGraph(string cEquation, double nFrom, double nTo, double
nStepInc, int nEquColor, bool lConnect, int nXCenter, int nYCenter, bool
lAddCoords, Form frmFormName, double nEquScale)
    {
        if (string.IsNullOrEmpty(cEquation) || frmFormName == null)
        {
            MessageBox.Show("This program requires multiple parameters.");
            return;
        }

        if (nEquScale == 0)
        {
            nEquScale = 1;
        }
    }
}
```

Source: Visual FoxPro  
Output Language: C#  
Procedure/Function: Closing 1-132  
Section: 1  
Vendor: OpenAI  
API Key: dcsi  
Model: gpt-3.5-turbo  
Tokens Used Today: 5,071  
Output Filename: cgraph\_Closing.cs

Convert Refresh Need Help?

By default, the VFP Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) Visual FoxPro selected as the Source Database Type and (3) C# selected as the destination language.

Depending upon the destination database selected on the main FmPro Migrator window, you could also see FileMaker Pro or Microsoft Access VBA selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them.

You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.

## Converting FoxPro 2.6 to Visual FoxPro

Status	ID	Size	Script Name
Completed	1	2441	pgraph.prg
Completed	2	4760	cgraph.prg
Completed	3	6788	main.prg
Not Started	4	14406	fdproc.prg
Not Started	5	20234	datepick.prg
Not Started	6	1207	frmanimation

```
set talk off
set stat off
set scor off
set cent on
set date brit
do while .t.
  clear
  @5,20 to 19,45 doub
  @6,25 say "MAIN MENU"
  @7,21 to 7,44 doub
  k=0
  @ 9,25 prompt "DATA ENTRY"
```

```
PRG
SET TALK OFF
SET STAT OFF
SET SCORE OFF
SET CENT ON
SET DATE BRIT

CREATE FORM frmMainMenu

WITH frmMainMenu
ADD OBJECT lblMainMenu TO 5,20 WIDTH 19 HEIGHT 25 ;
VALUE "MAIN MENU"
ADD OBJECT ineMainMenu TO 7,21 WIDTH 23 HEIGHT 1
ADD OBJECT cmdDataEntry TO 9,25 WIDTH 15 HEIGHT 1 ;
VALUE "DATA ENTRY"
ADD OBJECT cmdReportPrinting TO 11,25 WIDTH 15 HEIGHT 1 ;
VALUE "REPORT PRINTING"
ADD OBJECT cmdProcess TO 13,25 WIDTH 15 HEIGHT 1 ;
VALUE "PROCESS"
ADD OBJECT cmdExit TO 15,25 WIDTH 15 HEIGHT 1 ;
VALUE "EXIT"
ADD OBJECT cmdQuitSystem TO 17,25 WIDTH 15 HEIGHT 1 ;
VALUE "QUIT TO SYSTEM"
ENDWITH
```

Selecting FoxPro 2.6 as the Source Database type enables the conversion of FoxPro 2.6 code into Visual FoxPro code, including the creation of forms from the FoxPro 2.6 command line code. Performing this conversion makes use of the text-davinci-003 model. You will also need to break up the size of the script text manually, if there aren't any procedures/functions.

In order to prepare the FoxPro 2.6 project for import into FmPro Migrator, perform the following steps:

- 1) Install Visual FoxPro 9.
- 2) Make a copy of the FoxPro 2.6 project.
- 3) Open and convert the copied project in Visual FoxPro 9. This will upgrade all of the files including the DBF files. This is why you want to work with a copy of the project - because the DBF files won't be readable in FoxPro after conversion. And you might need to continue developing the existing FoxPro app while performing your conversion.
- 4) Once the FoxPro 2.6 project has been converted into Visual FoxPro 9, create a DBC and add all of the DBF files to the DBC.

5) Manually add 1 empty form to the Visual FoxPro 9 project.

Now you are ready to follow the existing PDF manual showing how to import the project into FmPro Migrator for conversion.

### VFP Code Conversion Workbench Demo



+



## VFP Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the VFP Code Conversion Workbench will open in Demo mode.

Apre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own Visual FoxPro project.

Clicking the Convert button will place the pre-converted script into the Converted Script field.

## Using the Microsoft Access Code Conversion Workbench - Access to FmPro Conversions

---

This chapter shows how to use the Microsoft Access Code Conversion Workbench - included with the AI Accelerated version of FmPro Migrator. This feature replaces copying the unconverted scripts into the FileMaker Pro database as was done previously. Before performing the code conversion with the Microsoft Access Code Conversion Workbench, you should have already imported the Microsoft Access database into FmPro Migrator, including tables, relationships, value lists, forms/reports and scripts.

### Access to FmPro Migration Button - GUI Tab

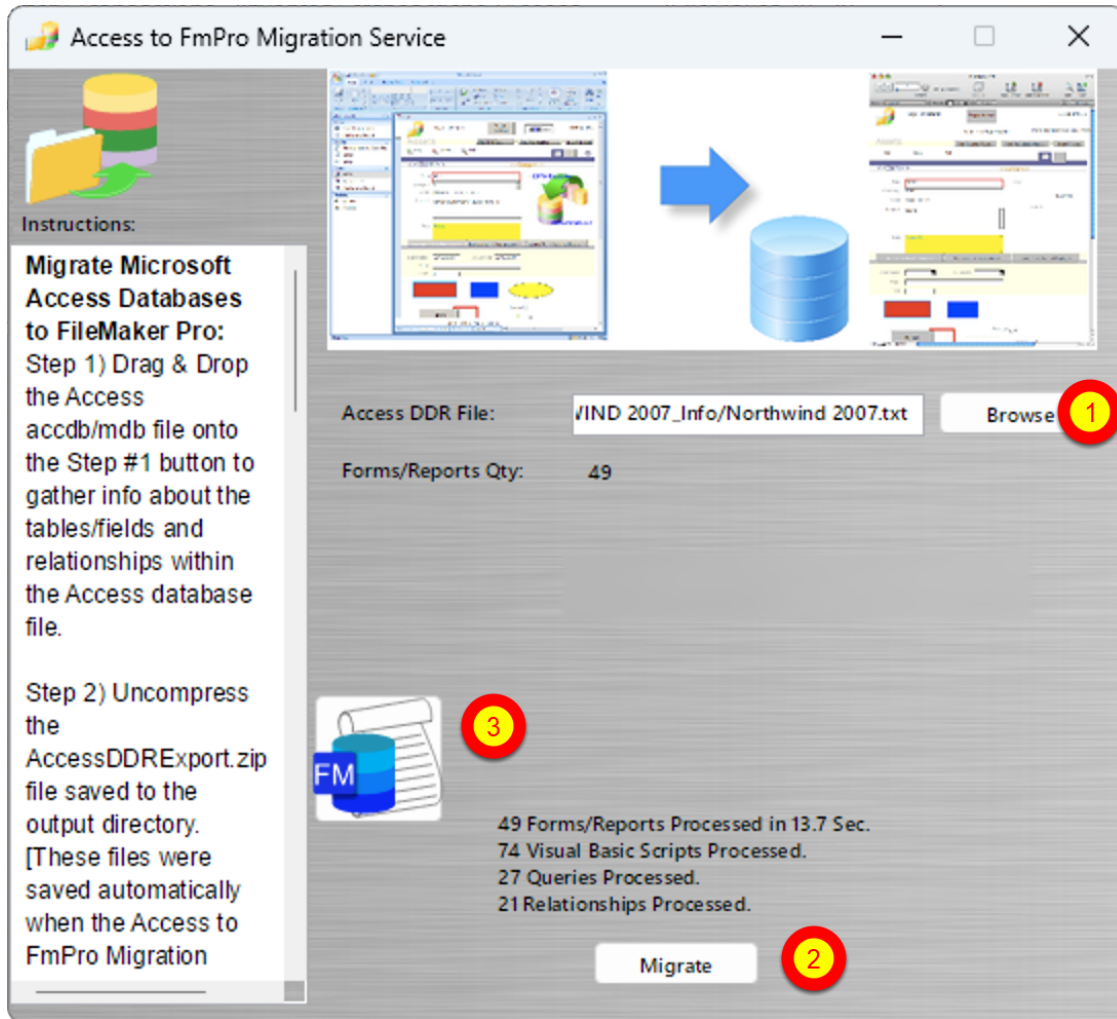


### Access to FmPro Migration

Click the [Access to FmPro Migration](#) button on the GUI tab of the Migration Process window of FmPro Migrator.



## Access to FmPro Migration Window



Prior to reaching this step, you should have, (1) selected the AccessDDRExport text file and (2) clicked the Migrate button to import the Access database metadata into FmPro Migrator. You may have already started building the new FileMaker Pro database which would mean that you are ready (3) to click the Code Conversion Workbench button.

## Microsoft Access Code Conversion Workbench Window

Code Conversion Workbench

# Microsoft Access Code Conversion Workbench

Completed: 2 of 6 Scripts Load Data

Status	ID	Size	Script Name
Completed	1	175	Open Script
Completed	2	46	Clear Sort Indicator
Completed	3	35	Toggle Window
Not Started	4	675	Find
Not Started	5	59	Go to Form Layout
Not Started	6	66	Go to Information Layout
Not Started	7	66	Go to List Layout
Not Started	8	354	Sort by Category
Not Started	9	333	Sort by Item
Not Started	10	339	Sort by Model
Not Started	11	381	Sort by Serial Number
Not Started	12	369	Sort by Assigned To
Not Started	13	355	Sort by Location
Not Started	14	228	Sort by Category Ascending
Not Started	15	229	Sort by Category Descending
Not Started	16	188	Sort by Item Ascending
Not Started	17	189	Sort by Item Descending
Not Started	18	190	Sort by Model Ascending
Not Started	19	191	Sort by Model Descending

**Source Script (Editable):** Size: 175

```
Module: Order Details:Form_Order Details
Option Compare Database
Option Explicit

Sub SetDefaultShippingAddress()
If IsNull(Me![Customer ID]) Then
ClearShippingAddress
Else
Dim rs As New RecordsetWrapper
If rs.OpenRecordset("Customers Extended", "[ID] = " & Me.Customer_ID)
```

**Command (Editable):**

Translate this Microsoft Access VBA code into FileMaker Pro script

**Converted Script:**

```
FileMaker Pro
Module: Order Details:Form_Order Details
#SetDefaultShippingAddress
If IsEmpty(GetField("Customer ID")) Then
Perform Script ["ClearShippingAddress"]
Else
Set Variable [rs; Value:New RecordsetWrapper]
If $rs.OpenRecordset("Customers Extended"; "[ID] = " & GetField("Customer ID")) Then
With $rs.Recordset
Set Field ["Ship Name"; $rs.Recordset.GetField("Contact Name")]
Set Field ["Ship Address"; $rs.Recordset.GetField("Address")]
Set Field ["Ship City"; $rs.Recordset.GetField("City")]
Set Field ["Ship State/Province"];
$rs.Recordset.GetField("State/Province")]
Set Field ["Ship ZIP/Postal Code"; $rs.Recordset.GetField("ZIP/Postal Code")]
Set Field ["Ship Country/Region";
$rs.Recordset.GetField("Country/Region")]
End With
End If
End If
```

Source: Microsoft Access

Output Language: FileMaker Pro

Convert

Vendor: OpenAI

API Key: dcsi

Model: gpt-3.5-turbo Refresh

Tokens Used Today: 597

Output Filename: Open Script.txt

Need Help?

By default, the Microsoft Access Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) Microsoft Access selected as the Source Database Type and (3) FileMaker Pro selected as the destination language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.



+



## Microsoft Access Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the Microsoft Access Code Conversion Workbench will open in Demo mode. A previously converted set of sample scripts will be displayed in the grid instead of the scripts from your own Microsoft Access database. Clicking any of the scripts will display the text "This demo only includes a few scripts." in the Source Script field.

[At the present time, there aren't any Microsoft Access demo scripts.]

## Using the FmPro Code Conversion Workbench - FmPro to Access Conversions

---

This chapter shows how to use the FmPro Code Conversion Workbench. Before performing the code conversion with the FmPro Code Conversion Workbench, you should have already imported the FileMaker Pro database into FmPro Migrator, including tables, relationships, value lists, layouts and scripts.

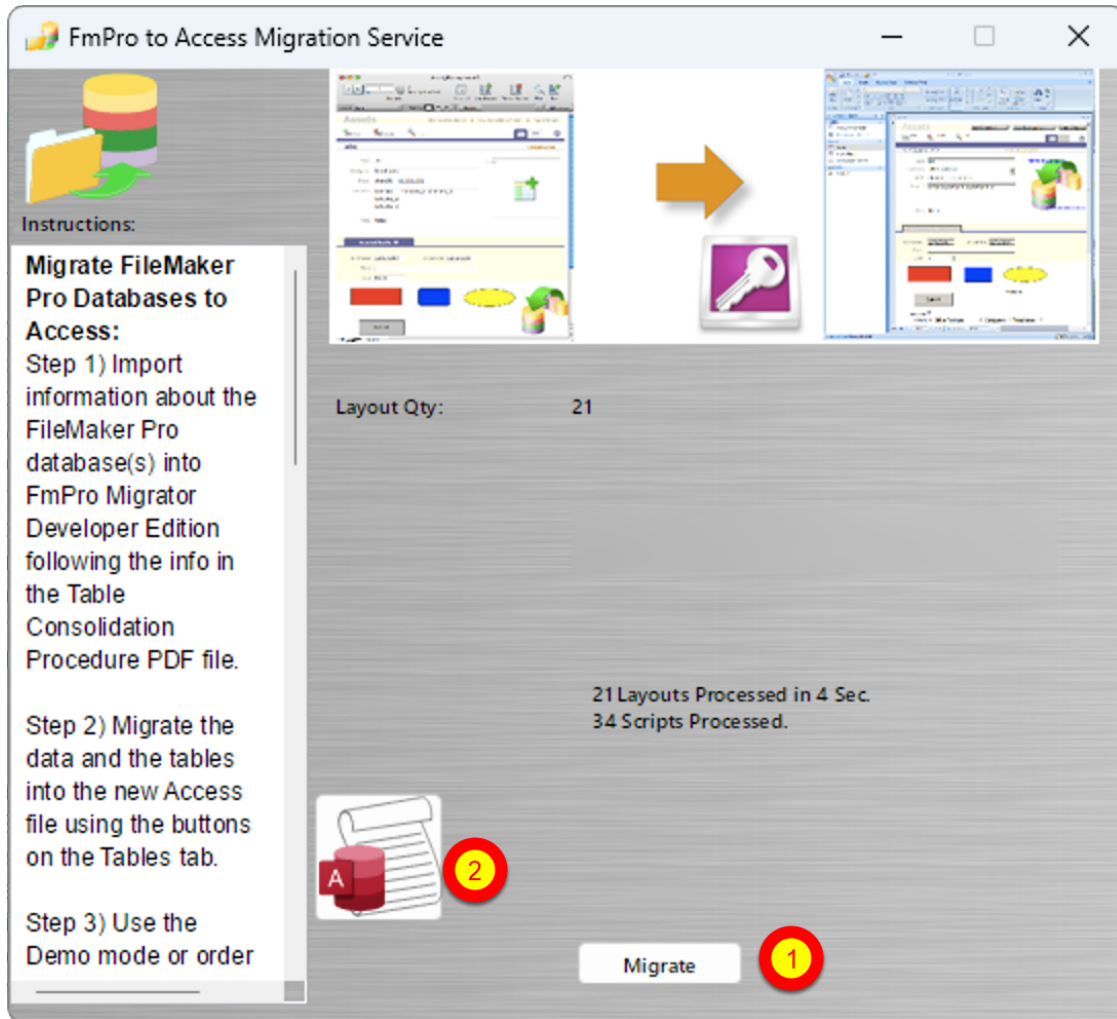
### FmPro to Access Migration Button - GUI Tab



### FmPro to Access Migration

Click the [FmPro to Access Migration](#) button on the GUI tab of the Migration Process window of FmPro Migrator.

## FmPro to Access Migration Window



Prior to reaching this step, you should have already imported the FileMaker Pro database into FmPro Migrator, (1) clicked the Migrate button to create the new Access database and run the `_LoadAllFormsAndReports.bas` script within the `AccessDBFiles` folder. After you have built the new Access database with all of the new forms/reports converted from the FileMaker Pro database you are ready (2) to click the Code Conversion Workbench button.

**Note:** Using the FmPro Code Conversion Workbench replaces the need to run the existing `_FmProConvertedScriptsVBA.bas` VBA script.

# FmPro Code Conversion Workbench Window

**Completed:** 2 of 6 Scripts Load Data

Status	ID	Size	Script Name
<input type="checkbox"/> Not Started	1	175	Open Script
<input type="checkbox"/> Not Started	2	46	Clear Sort Indicator
<input type="checkbox"/> Not Started	3	35	Toggle Window
<input checked="" type="checkbox"/> Completed	4	675	Find
<input type="checkbox"/> Not Started	5	59	Go to Form Layout
<input type="checkbox"/> Not Started	6	66	Go to Information Layout
<input type="checkbox"/> Not Started	7	66	Go to List Layout
<input type="checkbox"/> Not Started	8	354	Sort by Category
<input type="checkbox"/> Not Started	9	333	Sort by Item
<input type="checkbox"/> Not Started	10	339	Sort by Model
<input type="checkbox"/> Not Started	11	381	Sort by Serial Number
<input type="checkbox"/> Not Started	12	369	Sort by Assigned To
<input type="checkbox"/> Not Started	13	355	Sort by Location
<input type="checkbox"/> Not Started	14	228	Sort by Category Ascending
<input type="checkbox"/> Not Started	15	229	Sort by Category Descending
<input type="checkbox"/> Not Started	16	188	Sort by Item Ascending
<input type="checkbox"/> Not Started	17	189	Sort by Item Descending
<input type="checkbox"/> Not Started	18	190	Sort by Model Ascending
<input type="checkbox"/> Not Started	19	191	Sort by Model Descending

**Source Script (Editable):** Size: 675

```
Set Error Capture [ On ]
If [ Get(WindowMode) = 0 ]
# We're in Browse Mode: Enter Find Mode
Enter Find Mode [ ]
[ Pause ]
Else
# We're already in Find Mode. Let the Find button function the same as the
Continue button.
End If
Perform Find [ ]
If [ Get( LastError) = 400 ]
Show Custom Dialog [ Title: "Message"; Message: "No find criteria were entered."

```

**Command (Editable):**  
Translate this FileMaker Pro code into Microsoft Access VBA code

**Converted Script:**  
### Microsoft Access VBA  
On Error Resume Next  
If Application.CurrentObjectName = "" Then  
\*We're in Browse Mode: Enter Find Mode  
DoCmd.RunCommand acCmdFind  
If Err.Number = 2501 Then  
\*No find criteria were entered. All records will be displayed.  
MsgBox "No find criteria were entered. All records will be displayed.",  
vbInformation, "Message"  
DoCmd.RunCommand acCmdShowAllRecords  
End If  
Else  
\*We're already in Find Mode. Let the Find button function the same as the  
Continue button.  
DoCmd.RunCommand acCmdFind  
If Err.Number = 2501 Then  
\*No records match this request.  
MsgBox "No records match this request.", vbInformation, "Message"  
DoCmd.RunCommand acCmdShowAllRecords  
ElseIf DCount("", Application.CurrentObjectName) = 1 Then  
\*Only one record found  
DoCmd.RunCommand acCmdGoToRecord, , acGoTo, 1

Source: FileMaker Pro  
Output Language: Microsoft Access VBA  
Vendor: OpenAI  
API Key: dcsi  
Model: gpt-3.5-turbo  
Tokens Used Today: 1,073  
Output Filename: Find.bas  
Need Help?

By default, the FmPro Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) FileMaker Pro selected as the Source Database Type and (3) Microsoft Access VBA selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.



+



## FmPro Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the FmPro Code Conversion Workbench will open in Demo mode.

A pre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own FileMaker Pro database. Clicking any of the scripts will display the text "This demo only includes a few scripts." in the Source Script field.

[At the present time, there aren't any FileMaker Pro to Access demo scripts.]

## Using the FmPro Code Conversion Workbench - FmPro to Servoy Conversions

---

This chapter shows how to use the FmPro Code Conversion Workbench. Before performing the code conversion with the FmPro Code Conversion Workbench, you should have already imported the FileMaker Pro database into FmPro Migrator, including tables, relationships, value lists, layouts and scripts.

Just because this chapter of the manual is showing the conversion of FileMaker Pro scripts to Servoy, it is also possible to convert Visual FoxPro and Microsoft Access code to Servoy JavaScript.

### Servoy Migration Button - GUI Tab

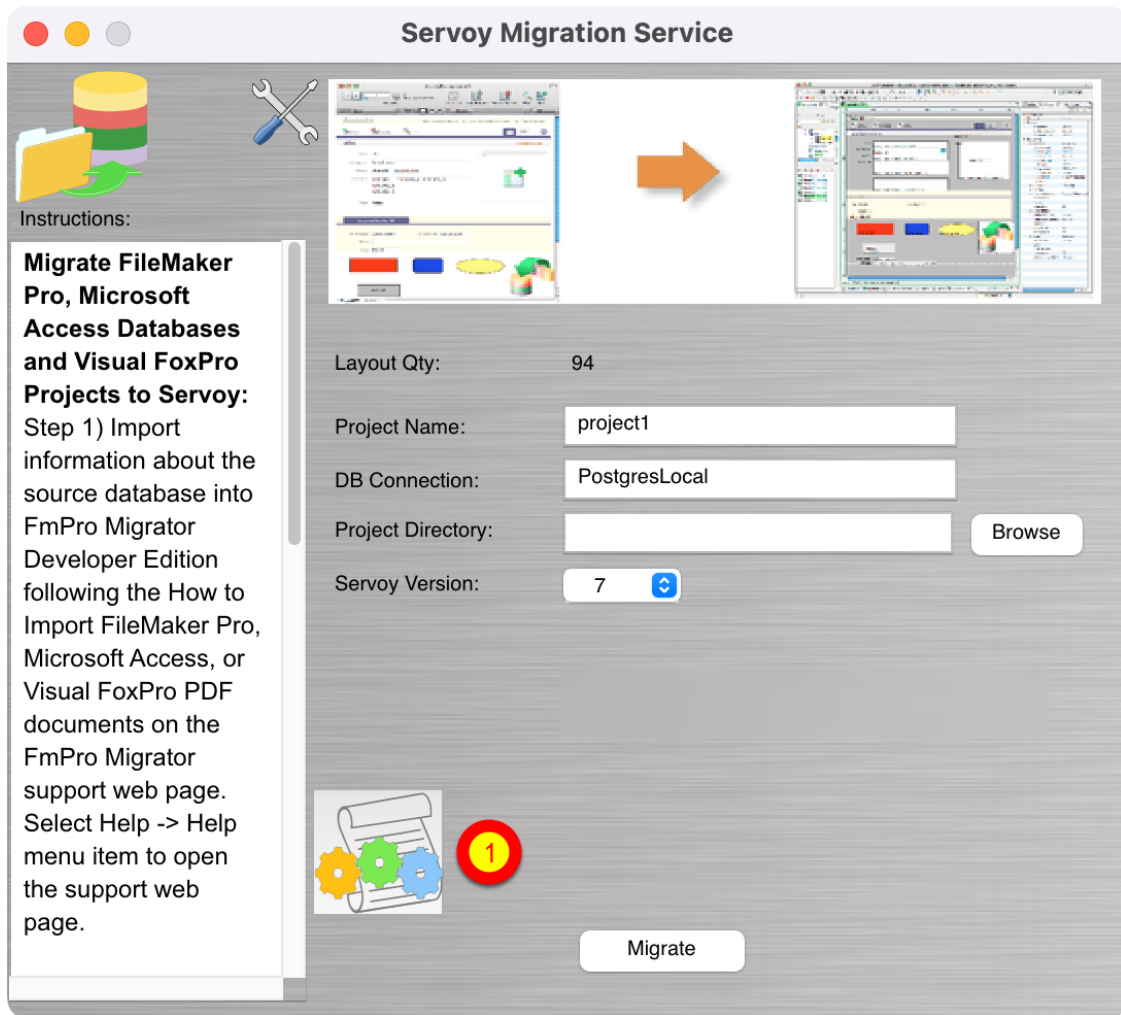


## Servoy Migration

Click the [Servoy Migration](#) button on the GUI tab of the Migration Process window of FmPro Migrator.



## Servoy Migration Window



Prior to reaching this step, you should have already imported the FileMaker Pro database into FmPro Migrator and created your Servoy project. It is not necessary to click the Migrate button on this window, because you could have created the Servoy project manually. You are now ready (1) to click the Code Conversion Workbench button.

## FmPro Code Conversion Workbench Window

Completed: 2 of 6 Scripts Load Data

Status	ID	Size	Script Name
<input type="checkbox"/> Not Started	1	175	Open Script
<input type="checkbox"/> Not Started	2	46	Clear Sort Indicator
<input type="checkbox"/> Not Started	3	35	Toggle Window
<input checked="" type="checkbox"/> Completed	4	675	Find
<input type="checkbox"/> Not Started	5	59	Go to Form Layout
<input type="checkbox"/> Not Started	6	66	Go to Information Layout
<input type="checkbox"/> Not Started	7	66	Go to List Layout
<input type="checkbox"/> Not Started	8	354	Sort by Category
<input type="checkbox"/> Not Started	9	333	Sort by Item
<input type="checkbox"/> Not Started	10	339	Sort by Model
<input type="checkbox"/> Not Started	11	381	Sort by Serial Number
<input type="checkbox"/> Not Started	12	369	Sort by Assigned To
<input type="checkbox"/> Not Started	13	355	Sort by Location
<input type="checkbox"/> Not Started	14	228	Sort by Category Ascending
<input type="checkbox"/> Not Started	15	229	Sort by Category Descending
<input type="checkbox"/> Not Started	16	188	Sort by Item Ascending
<input type="checkbox"/> Not Started	17	189	Sort by Item Descending
<input type="checkbox"/> Not Started	18	190	Sort by Model Ascending
<input type="checkbox"/> Not Started	19	191	Sort by Model Descending

**Source Script (Editable):** Size: 675

```
Set Error Capture [ On ]
If [ Get(WindowMode) = 0 ]
# We're in Browse Mode: Enter Find Mode
Enter Find Mode [ ]
[ Pause ]
Else
# We're already in Find Mode. Let the Find button function the same as the
Continue button.
End If
Perform Find [ ]
If [ Get(LastError) = 400 ]
Show Custom Dialog [ Title: "Message"; Message: "No find criteria were entered."

```

**Command (Editable):**

Translate this FileMaker Pro code into Servoy JavaScript code

**Converted Script:**

```
### Servoy JavaScript
// Set Error Capture [ On ]
databaseManager.setAutoSave(false);

if (application.getWindow().getModel() === 0) {
// We're in Browse Mode: Enter Find Mode
forms.myForm.foundset.find();
// [ Pause ]
forms.myForm.controller.pause();
} else {
// We're already in Find Mode. Let the Find button function the same as the
Continue button.
}

forms.myForm.foundset.loadAllRecords();

if (forms.myForm.foundset.getSize() === 0) {
plugins.dialogs.showInfoDialog("Message", "No records match this request.");
forms.myForm.foundset.loadAllRecords();
} else if (forms.myForm.foundset.getSize() === 1) {
forms.myForm.controller.show();
} else {
forms.myForm.controller.show();
forms.myForm.controller.loadRecords();
}

```

Source: FileMaker Pro

Output Language: Servoy JavaScript

Convert

Vendor: OpenAI

API Key: dcsi

Model: gpt-3.5-turbo Refresh

Tokens Used Today: 1,486

Output Filename: Find.js

Need Help?

By default, the FmPro Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) FileMaker Pro selected as the Source Database Type and (3) Servoy JavaScript selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.



+



## FMPro Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the FmPro Code Conversion Workbench will open in Demo mode.

A pre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own FileMaker Pro database. Clicking any of the scripts will display the text "This demo only includes a few scripts." in the Source Script field.

[At the present time, there aren't any FileMaker Pro to Servoy demo scripts.]

## Using the FmPro Code Conversion Workbench - FmPro to PHP Conversions

---

This chapter shows how to use the FmPro Code Conversion Workbench. Before performing the code conversion with the FmPro Code Conversion Workbench, you should have already imported the FileMaker Pro database into FmPro Migrator, including tables, relationships, value lists, layouts and scripts.

Just because this chapter of the manual is showing the conversion of FileMaker Pro scripts to PHP, it is also possible to convert Visual FoxPro and Microsoft Access code to PHP.

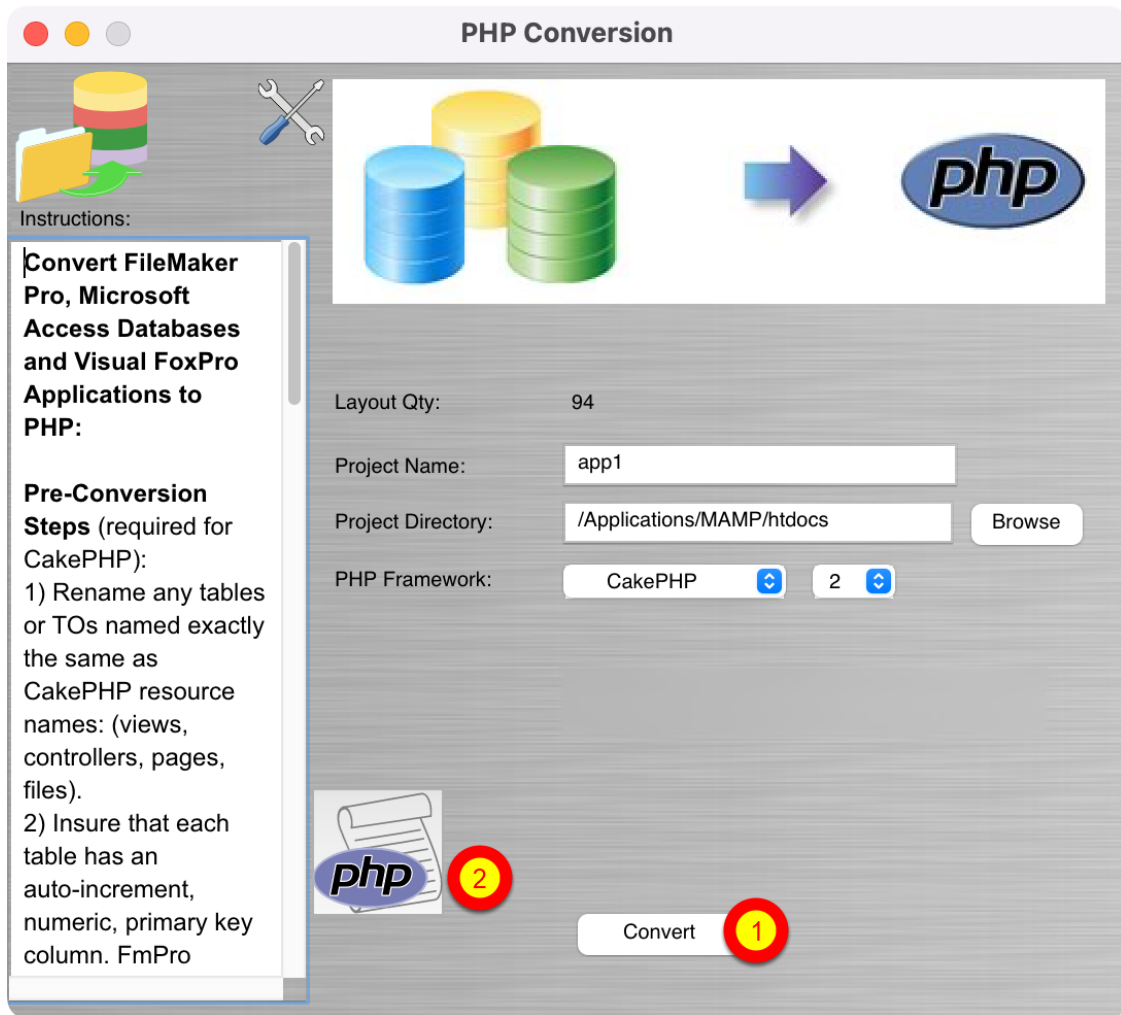
### PHP Migration Button - GUI Tab



### PHP Conversion

Click the [PHP Conversion](#) button on the GUI tab of the Migration Process window of FmPro Migrator.

## PHP Conversion Window



Prior to reaching this step, you should have already imported the FileMaker Pro database into FmPro Migrator, (1) created your PHP project with the Convert button. You are now ready to (2) click the Code Conversion Workbench button.

## FmPro Code Conversion Workbench Window

Status	ID	Size	Script Name
Not Started	1	175	Open Script
Not Started	2	46	Clear Sort Indicator
Not Started	3	35	Toggle Window
Completed	4	675	Find
Not Started	5	59	Go to Form Layout
Not Started	6	66	Go to Information Layout
Not Started	7	66	Go to List Layout
Not Started	8	354	Sort by Category
Not Started	9	333	Sort by Item
Not Started	10	339	Sort by Model
Not Started	11	381	Sort by Serial Number
Not Started	12	369	Sort by Assigned To
Not Started	13	355	Sort by Location
Not Started	14	228	Sort by Category Ascending
Not Started	15	229	Sort by Category Descending
Not Started	16	188	Sort by Item Ascending
Not Started	17	189	Sort by Item Descending
Not Started	18	190	Sort by Model Ascending
Not Started	19	191	Sort by Model Descending

```
Set Error Capture [ On ]
If [ Get(WindowMode) = 0 ]
# We're in Browse Mode: Enter Find Mode
Enter Find Mode [ ]
[ Pause ]
Else
# We're already in Find Mode. Let the Find button function the same as the
Continue button.
End If
Perform Find [ ]
If [ Get( LastError) = 400 ]
Show Custom Dialog [ Title: "Message"; Message: "No find criteria were entered.
```

```
### Laravel PHP
php
use Illuminate\Support\Facades\DB;

// Set error capture
DB::table('settings')->update(['error_capture' => true]);

if (request()->input('window_mode') == 0) {
// We're in Browse Mode: Enter Find Mode
DB::table('settings')->update(['find_mode' => true]);
// Pause for user input
// ...
} else {
// We're already in Find Mode. Let the Find button function the same as the
Continue button.
}

// Perform Find
$results = DB::table('records')->get();

if (count($results) == 0) {
// No records match this request
// Show custom dialog
```

By default, the FmPro Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) FileMaker Pro selected as the Source Database Type and (3) PHP selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.

In this example, you can see that Laravel has been written into the conversion prompt and that MySQL is the output database. This text is fully modifiable, giving the developer the option to enter any PHP framework or database server.



+



## FmPro Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the FmPro Code Conversion Workbench will open in Demo mode. A pre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own FileMaker Pro database. Clicking any of the scripts will display the text "This demo only includes a few scripts." in the Source Script field.

## Using the FmPro Code Conversion Workbench - FmPro to LiveCode Conversions

---

This chapter shows how to use the FmPro Code Conversion Workbench. Before performing the code conversion with the FmPro Code Conversion Workbench, you should have already imported the FileMaker Pro database into FmPro Migrator, including tables, relationships, value lists, layouts and scripts.

Just because this chapter of the manual is showing the conversion of FileMaker Pro scripts to LiveCode, it is also possible to convert Visual FoxPro and Microsoft Access code to LiveCode.

### LiveCode Conversion Button - GUI Tab

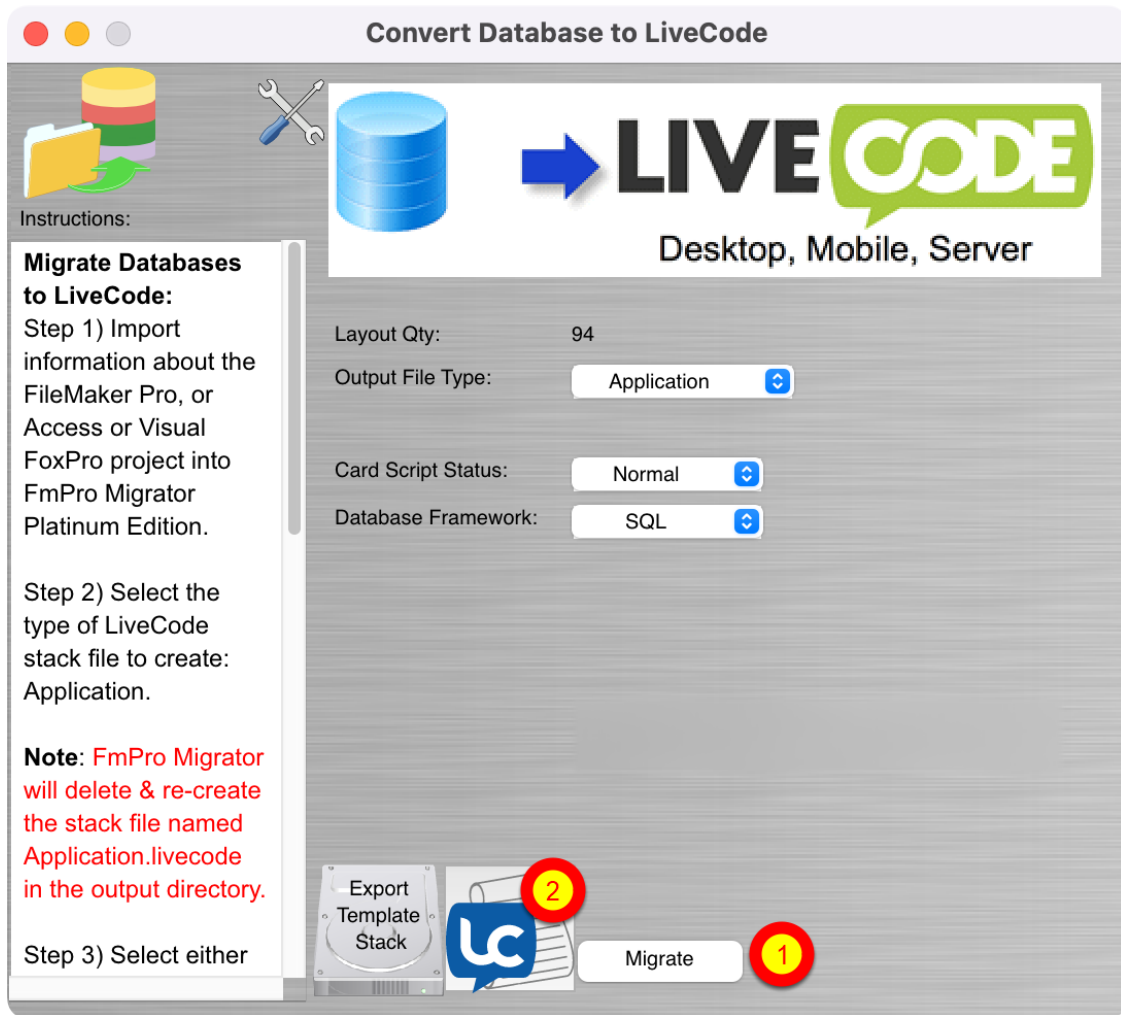


**LiveCode  
Conversion**

Click the [LiveCode Conversion](#) button on the GUI tab of the Migration Process window of FmPro Migrator.



## LiveCode Conversion Window



Prior to reaching this step, you should have already imported the FileMaker Pro database into FmPro Migrator, (1) created your new LiveCode stack with the Migrate button. You are now ready to (2) click the Code Conversion Workbench button.

## FmPro Code Conversion Workbench Window

Status	ID	Size	Script Name
<input type="checkbox"/> Not Started	1	175	Open Script
<input type="checkbox"/> Not Started	2	46	Clear Sort Indicator
<input type="checkbox"/> Not Started	3	35	Toggle Window
<input checked="" type="checkbox"/> Completed	4	675	Find
<input type="checkbox"/> Not Started	5	59	Go to Form Layout
<input type="checkbox"/> Not Started	6	66	Go to Information Layout
<input type="checkbox"/> Not Started	7	66	Go to List Layout
<input type="checkbox"/> Not Started	8	354	Sort by Category
<input type="checkbox"/> Not Started	9	333	Sort by Item
<input type="checkbox"/> Not Started	10	339	Sort by Model
<input type="checkbox"/> Not Started	11	381	Sort by Serial Number
<input type="checkbox"/> Not Started	12	369	Sort by Assigned To
<input type="checkbox"/> Not Started	13	355	Sort by Location
<input type="checkbox"/> Not Started	14	228	Sort by Category Ascending
<input type="checkbox"/> Not Started	15	229	Sort by Category Descending
<input type="checkbox"/> Not Started	16	188	Sort by Item Ascending
<input type="checkbox"/> Not Started	17	189	Sort by Item Descending
<input type="checkbox"/> Not Started	18	190	Sort by Model Ascending
<input type="checkbox"/> Not Started	19	191	Sort by Model Descending

```
Set Error Capture [ On ]
If [ Get(WindowMode) = 0 ]
# We're in Browse Mode: Enter Find Mode
Enter Find Mode [ ]
[ Pause ]
Else
# We're already in Find Mode. Let the Find button function the same as the
Continue button.
End If
Perform Find [ ]
If [ Get( LastError) = 400 ]
Show Custom Dialog [ Title: "Message"; Message: "No find criteria were entered.
```

```
### LiveCode
set the errorHandling to true
if the windowMode of this stack is 0 then
# We're in Browse Mode: Enter Find Mode
enter find mode
pause
else
# We're already in Find Mode. Let the Find button function the same as the
Continue button.
end if

perform find

if the result is "error" then
answer "No find criteria were entered. All records will be displayed." with "OK"
show all records
else
if the foundCount is 0 then
answer "No records match this request." with "OK"
show all records
else
if the foundCount is 1 then
go to layout "original layout"
else
```

By default, the FmPro Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) FileMaker Pro selected as the Source Database Type and (3) LiveCode selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.

In this example, you can see that LiveCode has been written into the conversion prompt and that MySQL is the output database. This text is fully modifiable giving the developer the option to enter any database server.



+



## FmPro Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the FmPro Code Conversion Workbench will open in Demo mode. A pre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own FileMaker Pro database. Clicking any of the scripts will display the text "This demo only includes a few scripts." in the Source Script field.  
[At the present time, there aren't any FileMaker Pro to LiveCode demo scripts.]

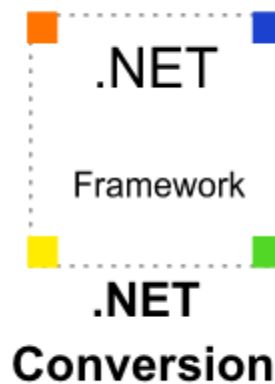
## Using the VFP Code Conversion Workbench - Visual FoxPro to .NET Conversions

---

This chapter shows how to use the VFP Code Conversion Workbench. Before performing the code conversion with the FmPro Code Conversion Workbench, you should have already imported the Visual FoxPro project into FmPro Migrator, including tables, relationships, value lists, forms/reports and scripts.

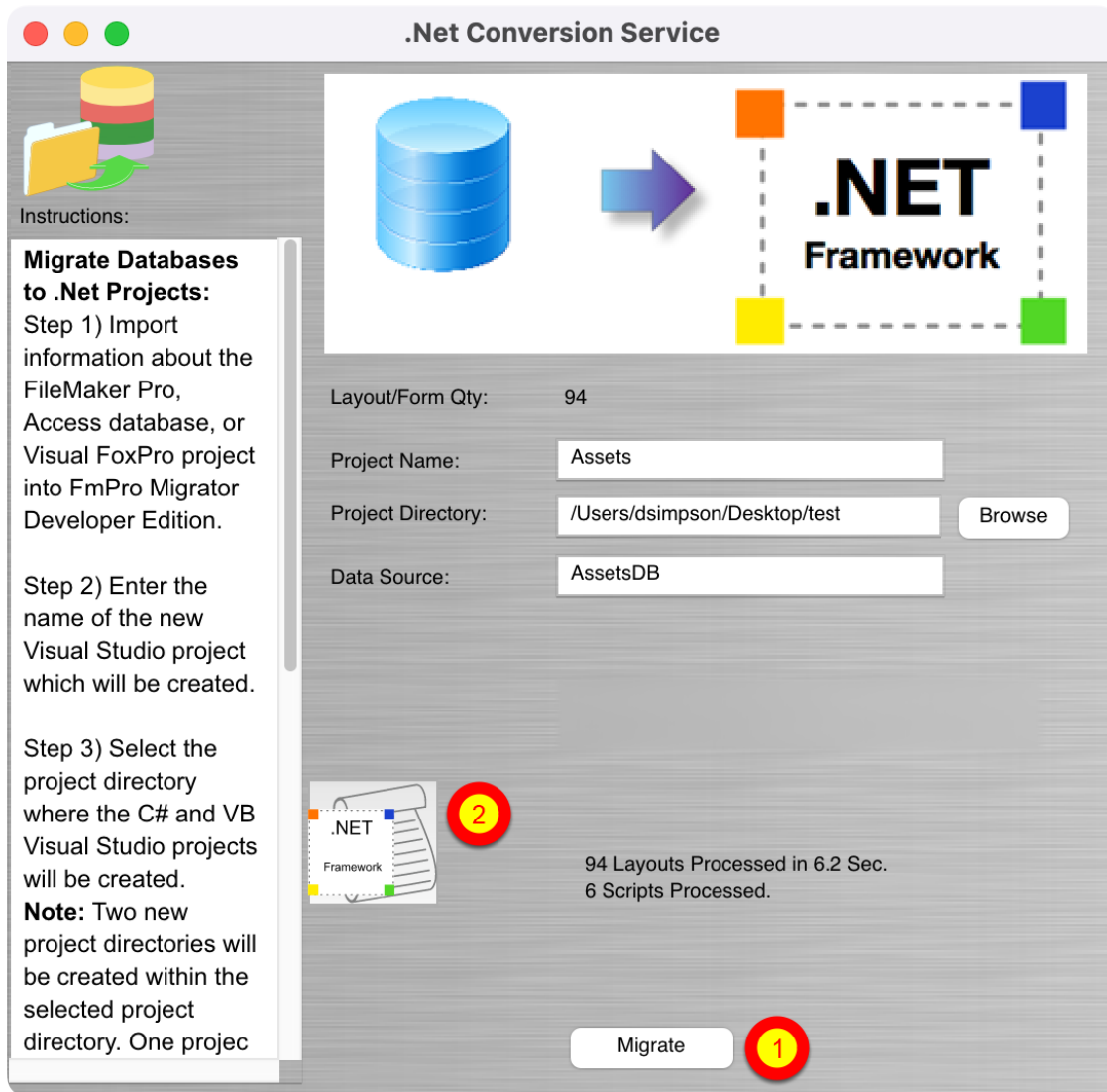
Just because this chapter of the manual is showing the conversion of Visual FoxPro scripts to .NET, it is also possible to convert FileMaker Pro and Microsoft Access code to .NET.

### LiveCode Conversion Button - GUI Tab



Click the [.NET Conversion](#) button on the GUI tab of the Migration Process window of FmPro Migrator.

## .NET Conversion Window



Prior to reaching this step, you should have already imported the Visual FoxPro project into FmPro Migrator using the Visual FoxPro Conversion button on the GUI tab, (1) created your new .NET project with the Migrate button. You are now ready to (2) click the Code Conversion Workbench button.

Code Conversion Workbench

# VFP Code Conversion Workbench

Completed: **2 of 6** Scripts Load Data

Status	ID	Size	Script Name
<input type="checkbox"/> Not Started	1	2441	pgraph.prg
<input type="checkbox"/> Not Started	2	4760	cgraph.prg
<input checked="" type="checkbox"/> Completed	3	6788	main.prg
<input type="checkbox"/> Not Started	4	14406	fdproc.prg
<input type="checkbox"/> Not Started	5	20234	datepick.prg
<input checked="" type="checkbox"/> Completed	6	1207	frmanimation

**Source Script (Editable):** Size: 976

```
PROCEDURE SolutionErrHandle
LOCAL InChoice
#DEFINE ERR_LOC "Error:"
#DEFINE FILEINUSE_LOC "A form cannot simultaneously be open in design mode and running."
#DEFINE FILEREADONLY_LOC "One of the necessary files is marked as read-only." + CHR(13) + "Please make sure you have read and write access to the file."

DO CASE
CASE ERROR() = 3 && File is in use
=MESSAGEBOX( ERR_LOC + MESSAGE() + CHR(13) + :
```

**Command (Editable):**

Translate this Visual FoxPro code into C# code

**Converted Script:**

```
...csharp
using System;

public class SolutionErrHandle
{
    public void HandleError()
    {
        int InChoice;
        const string ERR_LOC = "Error: ";
        const string FILEINUSE_LOC = "A form cannot simultaneously be open in design mode and running.";
        const string FILEREADONLY_LOC = "One of the necessary files is marked as read-only.\nPlease make sure you have read and write access to the file.";

        switch (Error())
        {
            case 3: // File is in use
                MessageBox.Show(ERR_LOC + Message() + "\n" + FILEINUSE_LOC, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                break;

            case 1718: // File is read-only
                MessageBox.Show(ERR_LOC + Message() + "\n" + FILEREADONLY_LOC, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
```

Source: Visual FoxPro

Output Language: C#

Procedure/Function: SolutionErrHandle 168-197

Convert

Vendor: OpenAI

API Key: dcsi

Model: gpt-3.5-turbo Refresh

Tokens Used Today: **3,238**

Output Filename: main\_SolutionErrHandle.cs

Need Help?

By default, the VFP Code Conversion Workbench window opens with the (1) scripts listed in the grid, (2) Visual FoxPro selected as the Source Database Type and (3) C# selected as the Output Language.

Just because the options are selected a certain way when the window opens, doesn't mean you can't change them. You can easily change the default settings on this window for Output Language, Vendor, API Key Type, or AI Model - and the changes will be saved into the project or application preferences for FmPro Migrator.



+



## VFP Code Conversion Workbench Demo

If you forgot to enter your license key or didn't have a license for the AI Accelerated version of FmPro Migrator, then the VFP Code Conversion Workbench will open in Demo mode.

A pre-converted set of sample scripts will be displayed in the grid instead of the scripts from your own Visual FoxPro project.

Clicking the Convert button will place the pre-converted script into the Converted Script field.

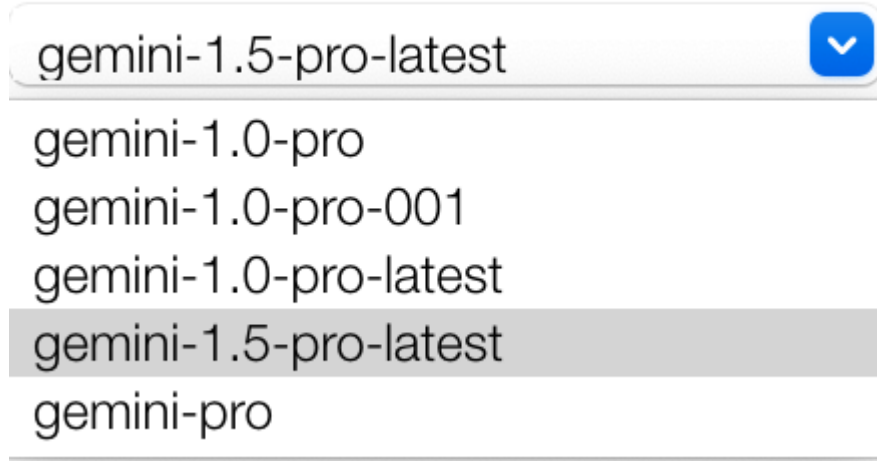
## Using Google Gemini Models

---

Google Gemini models are supported within the Code Code Conversion Workbench.

### Google Models

Model:



gemini-1.5-pro-latest

gemini-1.0-pro

gemini-1.0-pro-001

gemini-1.0-pro-latest

gemini-1.5-pro-latest

gemini-pro

When selecting Google as the AI Vendor, the gemini-1.0-pro model will be automatically selected. Click the Refresh button next to the model menu to see any updates to the list of available models.

Based upon testing experience, the gemini-1.0-pro models are not very useful for automated code conversion.

At the present time, the gemini-1.5-pro model is the best Google model to use for code conversion tasks. The 1.5-pro model is very close to the quality of the OpenAI gpt-4 model and it often returns results as much as 2x faster than gpt-4 model. So it is worth testing for your code conversion tasks.

#### **Notes:**

1) Since Google lists the gemini-1.5-pro model as being in Preview mode, there are rate limits currently in place (2 requests per minute, 1000 requests per hour). These rate limits will be lifted as the service expands.

2) The default model list is built into the Code Conversion Workbench during development, and the refreshed list is held in memory as long as the app is open.

3) Holding down the shift key when clicking the Refresh button will display a much longer list of models. These additional models are not functional for code conversion tasks and will return an error if you attempt to use them.



## Installing Ollama for Local LLM Usage

---

Ollama is a software package which manages downloading, managing and serving open source LLMs on your local computer.

Using Ollama to serve local LLMs is available with the FmPro Migrator Custom Development License.

### Installing Ollama



[Blog](#) [Discord](#) [GitHub](#)

[Models](#) [Sign in](#)

[Download](#)

### Download Ollama



Download for macOS

Requires macOS 11 Big Sur or later

While Ollama downloads, sign up to get notified of new updates.

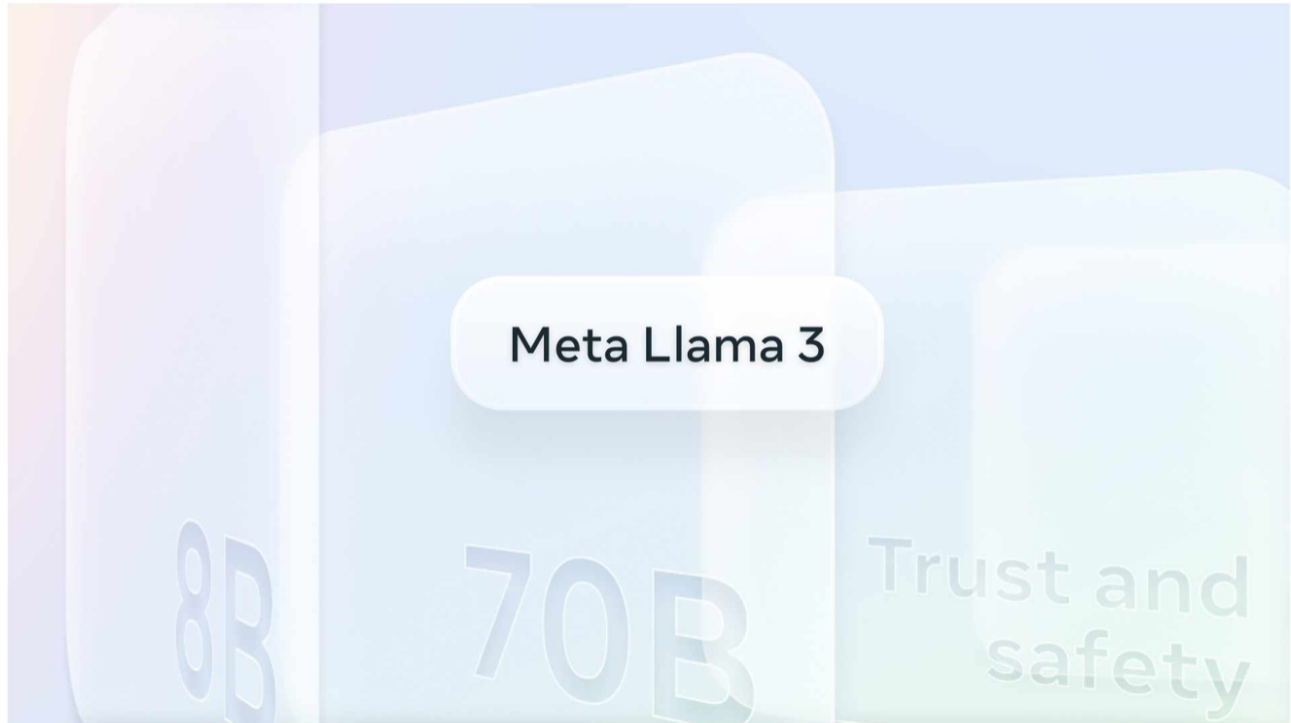
Get updates

Download and install Ollama from the [Ollama.com](https://ollama.com) website. Client software is available for macOS, Windows.

There are some considerations regarding the computer you choose for running local LLMs. Computers should have enough RAM to load the models into memory. Having one or more fast GPUs will also improve performance. A fast SSD will improve the performance of loading the model into memory.

# Llama 3

The most capable openly available LLM to date.



Meta Llama 3, a family of models developed by Meta Inc. are new state-of-the-art , available in both **8B** and **70B** parameter sizes (pre-trained or instruction-tuned).

Clicking the Models link at the top of the Ollama web page opens a list of available models.

Type the following command into a command prompt/terminal window to download a local copy of a model. For this example, the llama3 model is being used.

```
ollama run llama3:latest
```

This command loads a 4.7GB llama3 file with 8B parameters. The llama3 model generally seems to provide results comparable with OpenAI gpt-4 and Google Gemini 1.5-pro.

Note: Since you need enough RAM to load the model, I would recommend the 8B version of the model instead of the 70B version. The 8B file requires 16GB of RAM and 16GB of VRAM (on an

Apple Silicon based machine like an M2 MacBook Air, 24GB is enough memory because the unified architecture shares RAM between the CPU and GPUs without copying). If you wanted to use the 70B model, then you would need at least 140GB of RAM.

## Downloading Model - gemma:2b 3B



Note: this model requires Ollama 0.1.26 or later. [Download it here.](#)

Gemma is a new open model developed by Google and its DeepMind team. It's inspired by Gemini models at Google.

Gemma is available in both `2b` and `7b` parameter sizes:

- `ollama run gemma:2b`
- `ollama run gemma:7b` (default)

The models undergo training on a diverse dataset of web documents to expose them to a wide range of linguistic styles, topics, and vocabularies. This includes code to learn syntax and patterns of programming languages, as well as mathematical text to grasp logical reasoning.

The following command:

```
ollama run gemma:2b
```

Loads the Google 1.7GB gemma2b model with 3B parameters. This model loads and runs faster since it isn't as large and it gives you another option to test.

## Refreshing Models List

Converted Script:  
model 'codegemma:7b-code' not found, try pulling it first

API Key: dcsi

Model:  
codegemma:7b-code (9B)  
codegemma:7b-code (9B)  
gemma:2b (3B)  
llama3:latest (8B)  
datepick\_SolutionErrHandle.cs

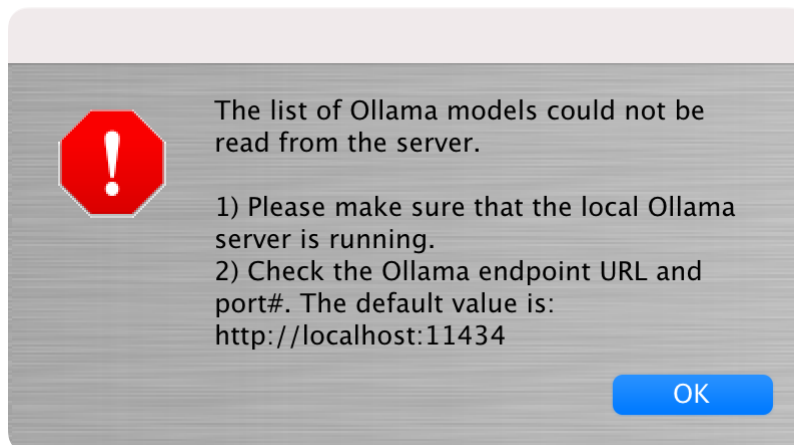
Refresh  
4,733

The first step to perform when using the Ollama models is to refresh the list of models. The default list represents the models built into FmPro Migrator during development, and won't reflect the actual models you have installed on your computer.

If you select an Ollama model which is not available, you will see an error like the following:

model 'codegemma:7b-code' not found, try pulling it first

## Ollama Localhost API Endpoint Connectivity



Problem Symptoms:

- 1) The models Refresh button displays the above error message because the Ollama server cannot be located.
- 2) The red error result text "Script Not Converted - Too Long?" will immediately be displayed above the Converted Script field as soon as the Convert button is clicked.

FmPro Migrator needs to have local internet access to the Ollama server endpoint when using Ollama machine learning models. The default URL for this local server is:

<http://localhost:11434>

Troubleshooting:

- 1) Make sure that the Ollama server is running. The server is started when launching the Ollama app, and a menu item will be displayed enabling control of the server. After the initial installation/configuration the app will quit as soon as it is launched - leaving the menu item.
- 2) You can go to the endpoint URL with a web browser and verify that you see the text: Ollama is running

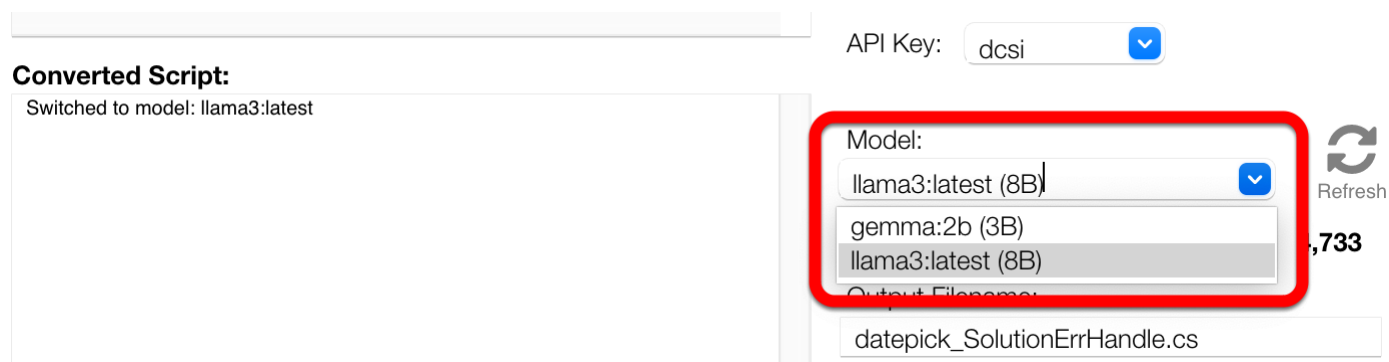
**Note:** When initially configured, "localhost" is the only address from which the Ollama server will respond. Using a different address requires updating the `/etc/systemd/system/ollama.service` configuration file on macOS

On Windows, the various parameters are configured with environment variables according to this article:

<https://docs.dify.ai/tutorials/model-configuration/ollama>

The localhost:11434 URL doesn't need to be entered into the Endpoint URL field, if it is left blank the default value shown above will be used automatically.

### Selecting an Ollama Model



Converted Script:  
Switched to model: llama3:latest

API Key: dcsi

Model:  
llama3:latest (8B)  
gemma:2b (3B)  
llama3:latest (8B)

Output Filename:  
datepick\_SolutionErrHandle.cs

Refresh

Ollama provides a way to load models into memory with the command:

```
ollama run <model name>
```

The Code Conversion Workbench GUI automates this task. Just select a model from the list and in a few seconds it will be loaded. The message:

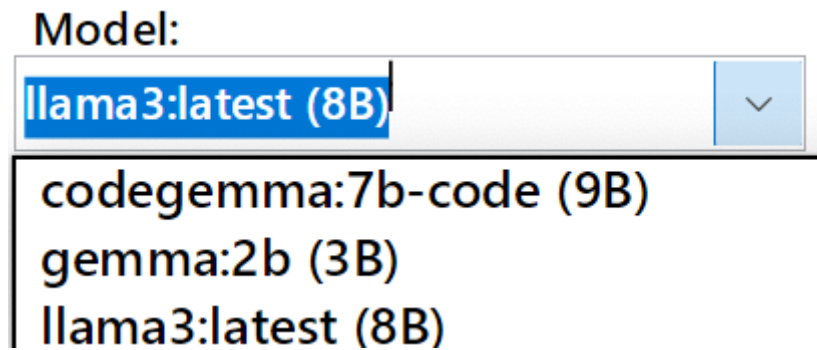
Switched to model: <model name>

will be displayed in the Converted script field.

From this point forward, you can select entire scripts or procedures/functions for conversion.

It is especially helpful to use the code splitting features of the Code Conversion Workbench window to reduce the size of the text being sent to the model to keep the sizes manageable.

### Ollama Models to Avoid



The codegemma:7b model is one of the default models built into the Model menu during development.

This model was used during testing and has not been found to be suitable for code conversion tasks - so please don't install it. Feel free to install other models suitable for your computer hardware if you like.

But to keep your selection process easy, the testing done during development has shown that the gemma:2b (3B) and llama3:latest (8B) models are working very well. Llama3 seems to work the best for code conversion and code explanation tasks, but since it is larger it takes a little longer to process requests. Llama3 generally seems to produce results at least as good as OpenAI gpt-3.5 and it is getting close to the quality of OpenAI gpt-4.

**Note:** Each Ollama supported model name consists of 2 pieces of information.

The left portion of the menu shows the name of the model as it has been installed locally.

The right portion of the menu (9B, 3B, 8B) shows the number of parameters used to train the model. Generally higher numbers are better. This way you can easily determine the

# Troubleshooting

### Script Too Large - Error Returned From Model

Source Script (Editable): **Script Too Large** Size: 20,234

```
***** datepick.prg
***** C:\DS\VFP_TESTS\SOLUTION\reports\datepick.prg
* Plot graph (Polar).
* Parameter:
* 1) cEquation (For radius) in terms of X. TYPE = Character.
* 2) nFrom. Where to stop counting for X. TYPE = Numeric.
* 3) nTo. Where to start counting for X. TYPE = Numeric.
* 4) nStepInc. Step increment. TYPE = Numeric.
* 5) nEquColor. TYPE = Numeric.
* 6) IConnect. If the previous point is connected to the cuurnt point with a line.
TYPE = Logical.
* 7) nXCenter. Point on form where x = 0. TYPE = Numeric.
```

Command (Editable):

Translate this Visual FoxPro code into C# code

Converted Script:

This model's maximum context length is 4097 tokens. However, your messages resulted in 6588 tokens. Please reduce the length of the messages.

This screenshot shows an example of a large script which exceeds the 4097 tokens available with the selected AI model. This message was returned from the server and it shows that there are 6589 tokens contained in the 20,234 characters of text sent to the model. **A token represents approximately 1 to 3 characters of text.** Fortunately, there are models available with a capacity of up to 16,000 tokens.

1) One potential solution is to use one of the 16K token capacity models like gpt-3.5-turbo-16k.



2) Another option is to try breaking up the script into individual procedures/functions. This can be advantageous even when a 16K model is available because sometimes the larger capacity models get overloaded.

### Script Not Converted - Too Long [Red Warning Text]

**Source Script (Editable):**

**Size:** 1,207

```
***** File: C:\DS\VFP_TESTS\SOLUTION\forms\graphics\anim.vcx
***** Object: franimation
PROCEDURE KeyPress
LPARAMETERS nKeyCode, nShiftAltCtrl
    this.drawmode = 10

ENDPROC
PROCEDURE MouseUp
LPARAMETERS nButton, nShift, nXCoord, nYCoord
IF THIS.Pendown
    thisform.line(this.oldx,this.oldy,this.currentx,this.currenty)
    this.drawmode = 1
```

**Command (Editable):**

Translate this Visual FoxPro code into C# code

**Converted Script:**

**Script Not Converted - Too Long**

This error is not returned from the server, it is displayed by the Code Conversion Workbench as a result of not receiving a response from the server. Therefore it is estimated by the software that the result could be due to the script being too long.

But it is also possible that the model was overloaded on the server. The gpt4 model was used for this test and experience has shown that the gpt4 model seems to get overloaded more often than the gpt-3.5-turbo model.

- 1) One possible solution is to break up the script into smaller amounts of text. Even though the script is reasonably sized, but since this script contains 4 individual procedures they could be converted separately - especially if you really require the additional code conversion quality of the gpt4 model.
- 2) Switching models from gpt4 to gpt-3.5-turbo is another option.
- 3) Trying again after a few minutes if the model is overloaded. Sometimes the models seem to work more quickly on weekends compared with during the work week.
- 4) Verify that your internet connection is functioning properly.

## OpenAI API Endpoint Connectivity

```
ping api.openai.com
PING api.openai.com (104.18.7.192): 56 data bytes
64 bytes from 104.18.7.192: icmp_seq=0 ttl=55 time=9.495 ms
64 bytes from 104.18.7.192: icmp_seq=1 ttl=55 time=7.441 ms

tracert api.openai.com
tracert: Warning: api.openai.com has multiple addresses; using 104.18.6.192
tracert to api.openai.com (104.18.6.192), 64 hops max, 40 byte packets
 1  10.1.0.1 (10.1.0.1)  2.069 ms  0.594 ms  0.543 ms
 2  192.168.1.254 (192.168.1.254)  2.502 ms
    76-244-40-1.lightspeed.sntcca.sbcglobal.net (76.244.40.1)  3.423 ms  6.513 ms
 3  71.148.149.96 (71.148.149.96)  6.950 ms  3.776 ms  3.759 ms
 4  12.242.117.22 (12.242.117.22)  5.548 ms  5.663 ms  7.857 ms
```

### Problem Symptoms:

- 1) The models menu Refresh button clears the list of models.
- 2) The busy indicator spins and doesn't return results or any error message after clicking the Convert button.

FmPro Migrator needs to have internet access to the OpenAI endpoint when using OpenAI machine learning models. This URL is:

[api.openai.com](https://api.openai.com)

### Troubleshooting:

Try performing a ping or traceroute command on the endpoint URL to insure that your computer can connect with the OpenAI API service.

If you cannot reach the endpoint URL, please check with your internet provider or corporate networking/security team for assistance.

Please see the section of the manual regarding running machine learning models locally with Ollama.

## Google Gemini API Endpoint Connectivity

```
ping generativelanguage.googleapis.com
PING generativelanguage.googleapis.com (142.250.189.170): 56 data bytes
64 bytes from 142.250.189.170: icmp_seq=0 ttl=57 time=6.527 ms
64 bytes from 142.250.189.170: icmp_seq=1 ttl=57 time=6.190 ms
64 bytes from 142.250.189.170: icmp_seq=2 ttl=57 time=6.321 ms

tracert generativelanguage.googleapis.com
tracert: Warning: generativelanguage.googleapis.com has multiple addresses; using 142.250.189.170
tracert to generativelanguage.googleapis.com (142.250.189.170), 64 hops max, 40 byte packets
 1  10.1.0.1 (10.1.0.1)  1.255 ms  0.535 ms  0.375 ms
 2  192.168.1.254 (192.168.1.254)  2.007 ms
    76-244-40-1.lightspeed.sntcca.sbcglobal.net (76.244.40.1)  2.905 ms  2.918 ms
```

### Problem Symptoms:

- 1) The models menu Refresh button clears the list of models.
- 2) The busy indicator spins and doesn't return results or any error message after clicking the Convert button.

FmPro Migrator needs to have internet access to the Google Gemini API endpoint when using Google machine learning models. This URL is:

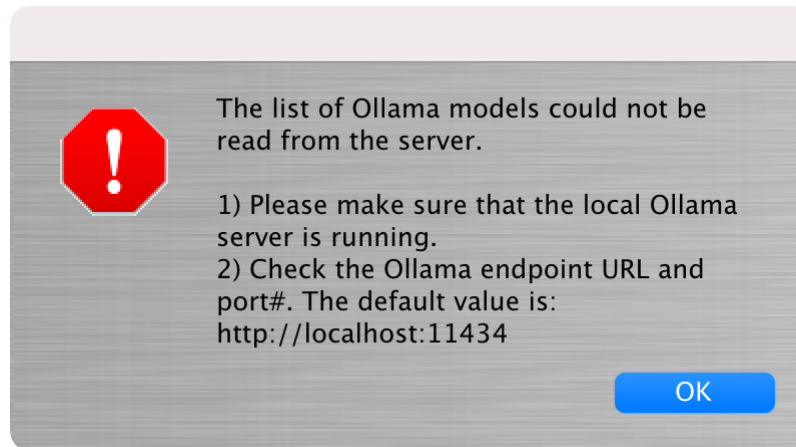
[generativelanguage.googleapis.com](https://generativelanguage.googleapis.com)

### Troubleshooting:

Try performing a ping or traceroute command on the endpoint URL to insure that your computer can connect with the Google Gemini API service.

If you cannot reach the endpoint URL, please check with your internet provider or corporate networking/security team for assistance.

Please see the section of the manual regarding running machine learning models locally with Ollama.



### Problem Symptoms:

- 1) The models Refresh button displays the above error message because the Ollama server cannot be located.
- 2) The red error result text "Script Not Converted - Too Long?" will immediately be displayed above the Converted Script field as soon as the Convert button is clicked.

FmPro Migrator needs to have local internet access to the Ollama server endpoint when using Google machine learning models. The default URL for this local server is:

<http://localhost:11434>

### Troubleshooting:

- 1) Make sure that the Ollama server is running. The server is started when launching the Ollama app, and a menu item will be displayed enabling control of the server. After the initial installation/configuration the app will quit as soon as it is launched - leaving the menu item.
- 2) You can go to the endpoint URL with a web browser and verify that you see the text: Ollama is running

**Note:** When initially configured, "localhost" is the only address from which the Ollama server will respond. Using a different address requires updating the `/etc/systemd/system/ollama.service` configuration file on macOS

On Windows, the various parameters are configured with environment variables according to this article:

<https://docs.dify.ai/tutorials/model-configuration/ollama>

The localhost:11434 URL doesn't need to be entered into the Endpoint URL field, if it is left blank the default value shown above will be used automatically.