

FmPro Migrator - FileMaker Pro, Microsoft Access, Visual FoxPro to .Net Automated Conversion Procedure



.com Solutions Inc.

1	Step 1 - Import Database Info - FileMaker Pro	
1.1	Importing FileMaker Pro Database Info	4
2	Step 1 - Import Database Info - Microsoft Access	
2.1	Importing Microsoft Access Database Info	6
3	Step 1 - Import Database Info - Visual FoxPro	
3.1	Importing Visual FoxPro Projects	8
4	.Net Conversion Processing	
4.1	Using Demo Mode - .Net Conversion Service	10
4.2	Using Licensed Mode - .Net Conversion Service	12
4.3	Review of Generated Visual Studio 2010 Project Files	17
5	.Net Conversion Service - Manual Tasks	
5.1	Create ADO.NET Data Source	21
5.2	Manual Tasks - .Net Conversion Service	29

Step 1 - Import Database Info - FileMaker Pro

Importing FileMaker Pro Database Info

1) Prior to performing any of the migration procedures listed below, it is recommended that you review the database structure of the FileMaker database(s) you are migrating. It is important to insure that each table is configured with a Primary Key column. FmPro Migrator looks for columns having the Unique and Not Empty validation properties in order to automatically determine which column should be created as a Primary Key column in the SQL database.

2) Download and following the instructions in the [How to Import FileMaker Pro Databases into FmPro Migrator](#) PDF manual from the FmPro Migrator support web page. Select Help from the Help menu in FmPro Migrator and your web browser will open this web page. It is generally a good idea to migrate the data into the SQL database, and also create relationships in the SQL database prior to migrating the Layouts into another development environment.

3) If you are transferring data from FileMaker Pro to a SQL database server, then download the appropriate manual on the support page for the destination SQL database. The [Pre-Migration Preparation Process](#) PDF provides another resource for migrating the data from FileMaker Pro to SQL database servers.

At the completion of these procedures, your data should already be migrated to the destination SQL database, and the Layouts, Value Lists, Scripts and Relationships should have been imported into FmPro Migrator.

Step 1 - Import Database Info - Microsoft Access

Importing Microsoft Access Database Info

1) Prior to performing any of the migration procedures listed below, it is recommended that you review the database structure of the Microsoft Access database(s) you are migrating. It is important to insure that each table is configured with a Primary Key column. FmPro Migrator looks for columns having the Unique and Not Empty validation properties in order to automatically determine which column should be created as a Primary Key column in the SQL database.

2) Download and following the instructions in the [How to Import Microsoft Access Databases into FmPro Migrator](#) PDF manual from the FmPro Migrator support web page. Select Help from the Help menu in FmPro Migrator and your web browser will open this web page.

3) If you are transferring data from Microsoft Access to a SQL database server, then you will likely use an importing procedure to import the data from the Access .mdb/.accdb file(s) into the destination SQL database. [FmPro Migrator supports directly copying data to FileMaker Pro databases, but not into SQL database servers.]

At the completion of these procedures, your data should already be migrated into the destination SQL database, and the Forms/Reports, Value Lists, Scripts and Relationships should have been imported into FmPro Migrator.

Step 1 - Import Database Info - Visual FoxPro

Importing Visual FoxPro Projects

1) Prior to performing any of the migration procedures listed below, it is recommended that you review the database structure of the Visual FoxPro project you are migrating. It is important to insure that each table is configured with a Primary Key column. FmPro Migrator looks for columns having the Unique and Not Empty validation properties in order to automatically determine which column should be created as a Primary Key column in the SQL database. Also, columns marked as NOT NULL should not contain NULL values, or the data transfer process will fail.

2) Download and following the instructions in the [How to Import Visual FoxPro Projects into FmPro Migrator](#) PDF manual from the FmPro Migrator support web page. Select Help from the Help menu in FmPro Migrator and your web browser will open this web page.

3) If you are transferring data from Microsoft Access to a SQL database server, then you may use an importing procedure to import the data from the DBF file(s) into the destination SQL database. Or you may use FmPro Migrator to transfer data from the DBF files into the destination SQL database server.

At the completion of these procedures, your data should already be migrated into the destination SQL database, and the Forms/Reports, Value Lists, Scripts and Relationships should have been imported into FmPro Migrator.

.Net Conversion Processing

Using Demo Mode - .Net Conversion Service

FmPro Migrator includes a Demo mode for the .Net Conversion Service. Demo mode enables .Net developers to fully test the migration capabilities of the migration service with a limited number of layouts and scripts.

Demo mode also enables .Net developers to quickly create "Proof of Concept" conversion projects for prospective clients who need to migrate their existing database files.

Click Migrate Button

The screenshot shows the ".Net Conversion Service" application window. On the left, there is a sidebar with instructions for migrating databases to .Net projects. The main area contains a diagram of a database cylinder pointing to a ".NET Framework" box. Below this, there are input fields for "Project Name" (dcsl_ins), "Project Directory" (C:/ds/DCSI_INS_Test), and "Data Source" (dcsl_ds). The "Processing Type" is set to "Demo". A "Migrate" button is at the bottom. A summary box indicates "5 Layouts Processed in 6.5 Sec. 5 Scripts Processed.".

.Net Conversion Service

Instructions:

Migrate Databases to .Net Projects:

Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project

Layout/Form Qty: 87

Project Name: **1** dcsl_ins

Project Directory: C:/ds/DCSI_INS_Test **2** Browse

Data Source: **3** dcsl_ds

Processing Type: Demo Order License Key

QUOTE

5 Layouts Processed in 6.5 Sec.
5 Scripts Processed. **5**

Migrate **4**

- (1) Enter a name for the new .Net 4 Visual Studio project which will be created by FmPro Migrator.
- (2) Select an output directory where the generated project files will be written. A new folder having the same name as the project will be created within the selected Project Directory folder. Two folders will be created within this folder, for the VB and C# project files.

- (3) Enter a name for the ADO.NET data source which will be created manually within Visual Studio. This name will be used for naming the Entity Framework model within the project.
- (4) Click the Migrate button.
- (5) The new .Net project will be created and the migration statistics will be displayed. Since Demo mode is being used, only 5 Layouts/Scripts were processed.

Using Licensed Mode - .Net Conversion Service

Using the .Net Conversion Service in Demo mode, limits processing tasks to 5 Forms/Reports and 5 scripts. Ordering a license key lifts these operating limits for this service. Using the License Key allows for the processing of an unlimited number of database files and scripts for the purchased Layouts/Forms quantity during the duration of the license key.

Select Processing Type - Licensed

.Net Conversion Service

Instructions:

Migrate Databases to .Net Projects:

Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project

Layout/Form Qty: 87

Project Name: dcsi_ins

Project Directory: C:/ds/DCSI_INS_Test

Data Source: dcsi_ds

Processing Type: Demo

Order License Key

Migrate

- (1) Enter a name for the new .Net 4 Visual Studio project which will be created by FmPro Migrator.
- (2) Select an output directory where the generated project files will be written. A new folder having the same name as the project will be created within the selected Project Directory folder. Two folders will

be created within this folder, for the VB and C# project files.

(3) Enter a name for the ADO.NET data source which will be created manually within Visual Studio. This name will be used for naming the Entity Framework model within the project.

(4) Click the Order License Key button. FmPro Migrator will open the .com Solutions Inc. web store hosted by Kagi. The specified Forms/Reports quantity (5) and .Net Conversion service will automatically be added to the Kagi shopping cart.

Obtaining a Price Quote

The screenshot shows the ".Net Conversion Service" application window. On the left, there is an "Instructions:" panel with the following text:

Migrate Databases to .Net Projects:
Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project

The main area of the window displays a diagram of a database cylinder pointing to a ".NET Framework" box. Below this, the following fields are visible:

- Layout/Form Qty: 87
- Project Name: dcsi_ins
- Project Directory: C:/ds/DCSI_INS_Test (with a "Browse" button)
- Data Source: dcsi_ds
- Processing Type: Demo (with a dropdown arrow)

At the bottom right, there is an "Order License Key" button. At the bottom center, there is a "Migrate" button. A red box highlights a "QUOTE" button with a green dollar sign icon, located in the bottom left area of the main window.

The Quote button links to a web form which creates a printable price quote & cost justification document based upon the number of Forms/Reports imported into FmPro Migrator Developer Edition. This document is designed to be suitable for project budget planning purposes and review by corporate

finance departments.

Entering a License Key

.Net Conversion Service

Instructions:

Migrate Databases to .Net Projects:

Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project


Layout/Form Qty: 87


Project Name: dcsi_ins

Project Directory: C:/ds/DCSI_INS_Test

Data Source: dcsi_ds

Processing Type:

License Key: 

QUOTE 

Once the payment processing has been completed on the Kagi web store, an email receipt containing the License Key for the .Net Conversion Service will automatically be sent within a few minutes.

Select Licensed from the Processing Type menu.

Copy the license key within the email receipt, then click the clipboard icon.

Validating the License Key

The screenshot shows the ".Net Conversion Service" window. On the left, an "Instructions" pane lists three steps for migrating databases to .Net projects. The main area displays a diagram of a database cylinder pointing to a ".NET Framework" box. Below this, form fields are populated with "Layout/Form Qty: 87", "Project Name: dcsi_ins", "Project Directory: C:/ds/DCSI_INS_Test", and "Data Source: dcsi_ds". The "Processing Type" is set to "Licensed". The "License Key" field is empty, but a red-bordered box displays the validation result: "Valid License Key" and "Valid Until 1/15/2011 (87 Forms)". A "Migrate" button is at the bottom right, and a "QUOTE" icon with a dollar sign is at the bottom left.

.Net Conversion Service

Instructions:

Migrate Databases to .Net Projects:

Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project


Layout/Form Qty: 87

Project Name: dcsi_ins


Project Directory: C:/ds/DCSI_INS_Test

Data Source: dcsi_ds

Processing Type: Licensed

License Key: 

Valid License Key
Valid Until 1/15/2011 (87 Forms)

 **QUOTE**

FmPro Migrator copies the license key from the clipboard, validates the key with the license key server via the internet and then displays the Forms/Reports quantity and expiration date for the license key.

.Net Conversion Service

Instructions:

Migrate Databases to .Net Projects:

Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.

Step 2) Enter the name of the new Visual Studio project which will be created.

Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.

Note: Two new project directories will be created within the selected project

Layout/Form Qty: 87

Project Name: dcsi_ins

Project Directory: C:/ds/DCSI_INS_Test Browse

Data Source: dcsi_ds

Processing Type: Licensed

License Key: MDEwODcf/Jc+IR+AHxo/Cv3g5jnXdb3RDC/\VZJ6DBxVDe1dschnsaW1QJLCR\Kty0qCJ\w!

Valid License Key
Valid Until 1/15/2011 (87 Forms)

QUOTE

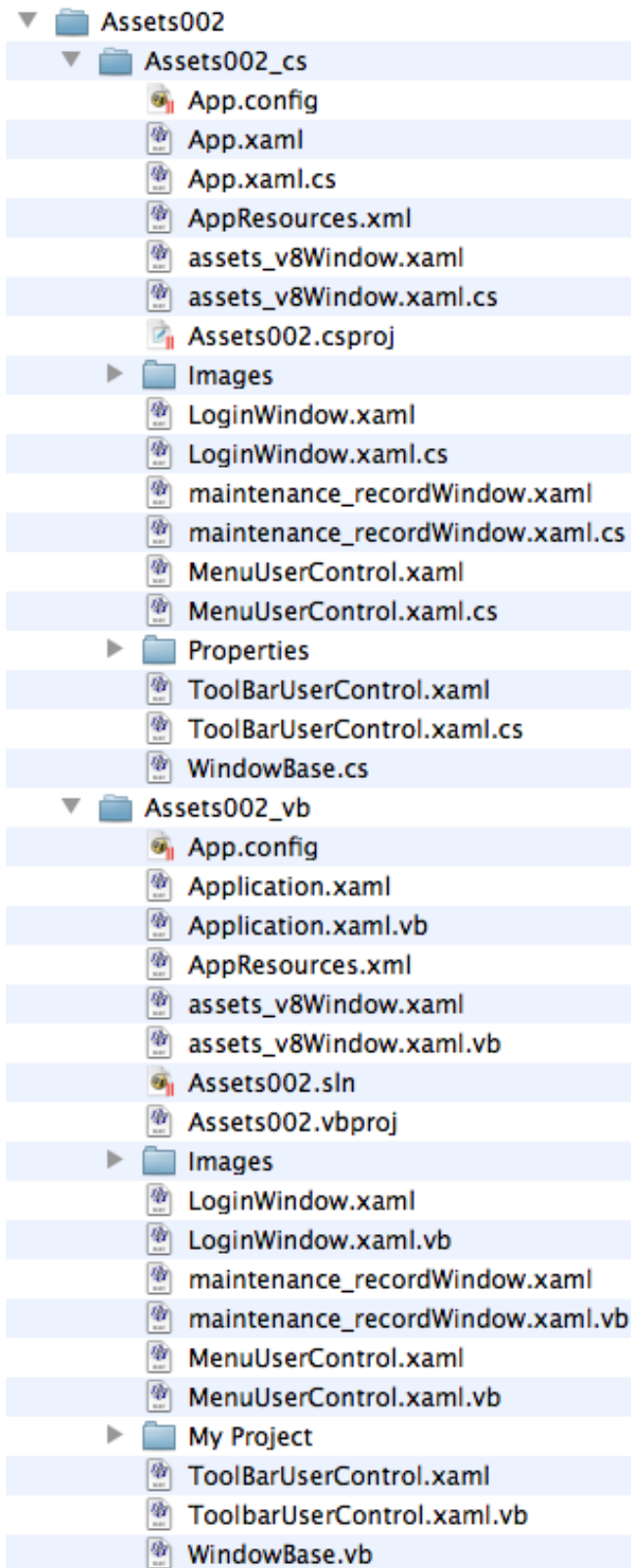
2 87 Layouts Processed in 16.6 Sec.
147 Scripts Processed.

1 Migrate

(1) Clicking the Migrate button with the license key enables the conversion of the number of Forms/Reports specified in the license key and the conversion of an unlimited number of scripts, with the (2) resulting processing statistics displayed below the license key field.

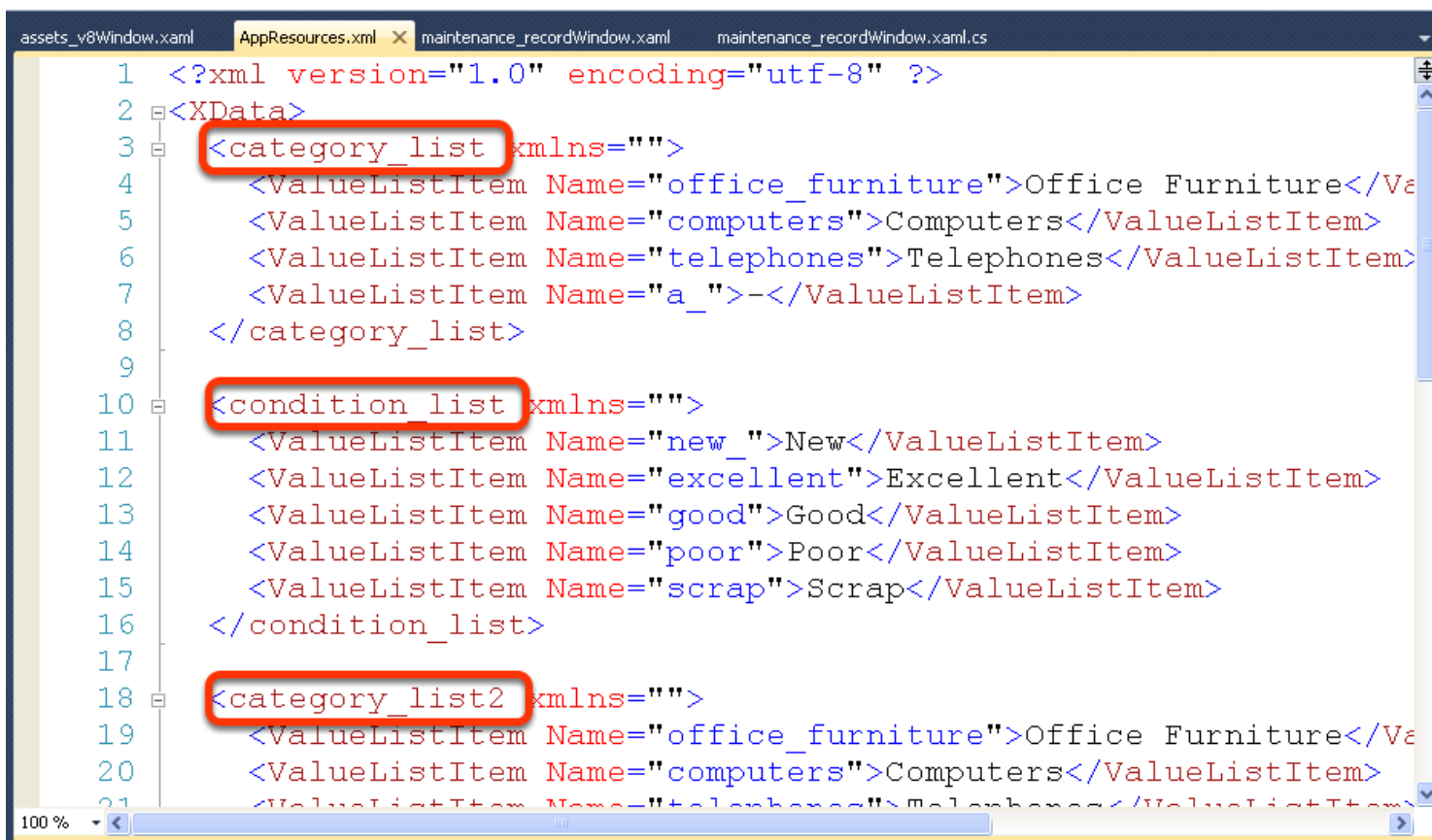
Review of Generated Visual Studio 2010 Project Files

.Net Project Files



A screenshot of an example project generated by FmPro Migrator is show above.

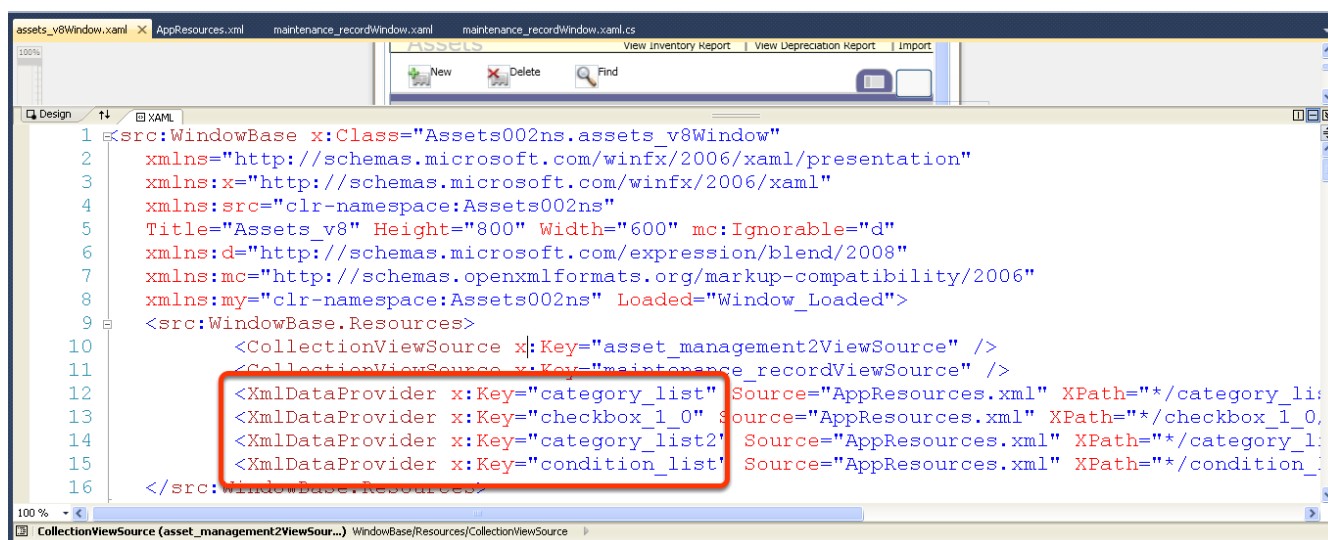
AppResources.xml Contents

A screenshot of a text editor showing the contents of the AppResources.xml file. The file is opened in a window titled 'AppResources.xml'. The code is XML and contains three main sections: a 'category_list', a 'condition_list', and a 'category_list2'. Each section is enclosed in a root tag with 'xmlns=""'. The 'category_list' and 'category_list2' sections contain 'ValueListItem' tags with 'Name' attributes and text values. The 'condition_list' section also contains 'ValueListItem' tags with 'Name' attributes and text values. The 'category_list2' section is partially visible at the bottom.

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <XData>
3   <category_list xmlns="">
4     <ValueListItem Name="office_furniture">Office Furniture</ValueListItem>
5     <ValueListItem Name="computers">Computers</ValueListItem>
6     <ValueListItem Name="telephones">Telephones</ValueListItem>
7     <ValueListItem Name="a_">-</ValueListItem>
8   </category_list>
9
10  <condition_list xmlns="">
11    <ValueListItem Name="new_">New</ValueListItem>
12    <ValueListItem Name="excellent">Excellent</ValueListItem>
13    <ValueListItem Name="good">Good</ValueListItem>
14    <ValueListItem Name="poor">Poor</ValueListItem>
15    <ValueListItem Name="scrap">Scrap</ValueListItem>
16  </condition_list>
17
18  <category_list2 xmlns="">
19    <ValueListItem Name="office_furniture">Office Furniture</ValueListItem>
20    <ValueListItem Name="computers">Computers</ValueListItem>
21    <ValueListItem Name="telephones">Telephones</ValueListItem>
```

Static Value Lists are converted into XML code which is saved in the AppResources.xml file. The Value List names displayed in the FmPro Migrator Value List tab are used as the identifier of each Value List. The Value List items follow the name tag, and contain a Name tag for each item followed by its data.

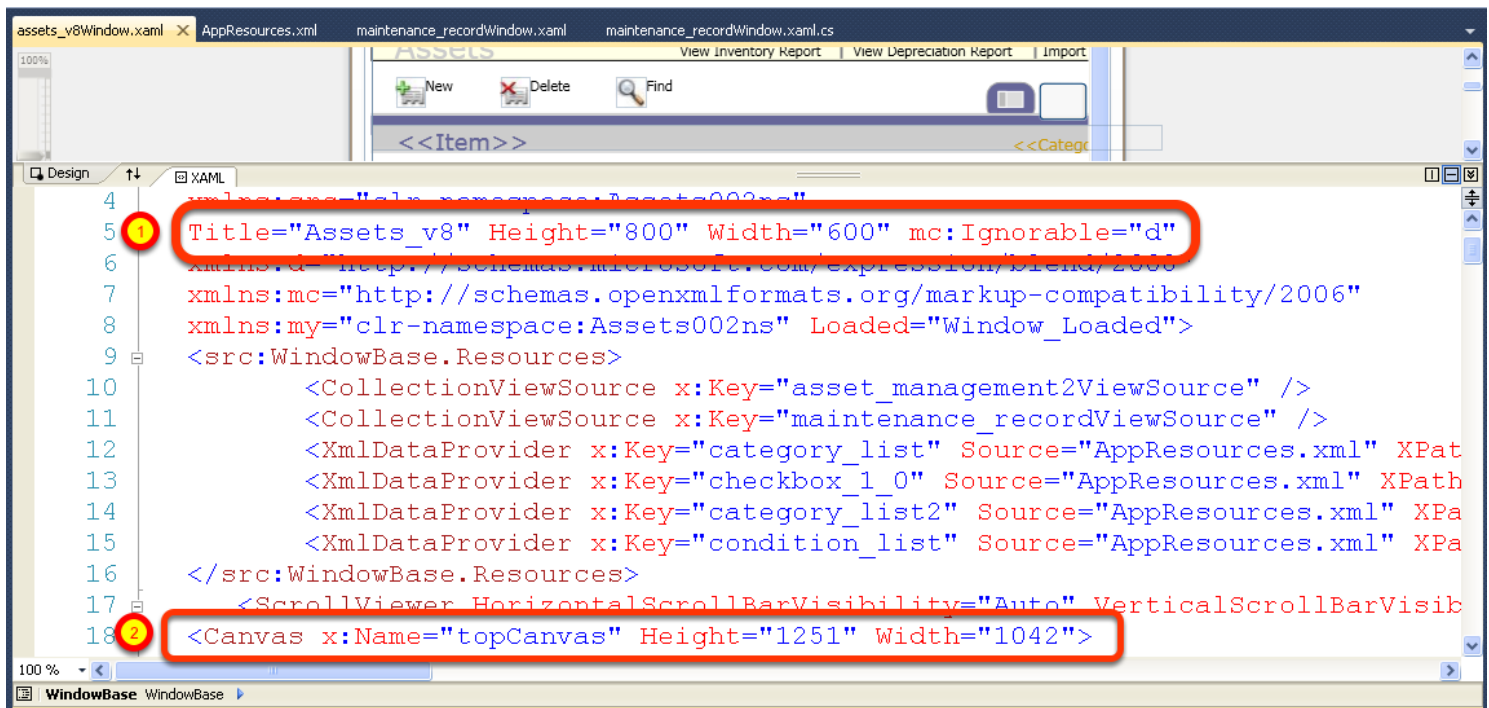
Value List Definition - Window XAML Code

A screenshot of a text editor showing the XAML code for a window. The code is in a window titled 'Assets'. The XAML code defines a 'WindowBase' with various properties and a 'CollectionViewSource' for 'asset_management2ViewSource'. The 'CollectionViewSource' is defined with 'XmlDataProvider' elements for 'category_list', 'checkbox_1_0', 'category_list2', and 'condition_list'. The 'category_list' and 'category_list2' providers are highlighted with a red box. The 'condition_list' provider is also highlighted with a red box.

```
1 <src:WindowBase x:Class="Assets002ns.assets_v8Window"
2   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4   xmlns:src="clr-namespace:Assets002ns"
5   Title="Assets_v8" Height="800" Width="600" mc:Ignorable="d"
6   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
7   xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
8   xmlns:my="clr-namespace:Assets002ns" Loaded="Window_Loaded">
9   <src:WindowBase.Resources>
10     <CollectionViewSource x:Key="asset_management2ViewSource" />
11     <CollectionViewSource x:Key="maintenance_recordViewSource" />
12     <XmlDataProvider x:Key="category_list" Source="AppResources.xml" XPath="*/category_list" />
13     <XmlDataProvider x:Key="checkbox_1_0" Source="AppResources.xml" XPath="*/checkbox_1_0" />
14     <XmlDataProvider x:Key="category_list2" Source="AppResources.xml" XPath="*/category_list2" />
15     <XmlDataProvider x:Key="condition_list" Source="AppResources.xml" XPath="*/condition_list" />
16   </src:WindowBase.Resources>
```

Each Value List used within a window, is defined as an XmlDataProvider near the top of the window XAML code file.

Window Scrollbar Implementation



Windows which contain objects beyond the displayed window size, will automatically show Horizontal or Vertical scrollbars on an as needed basis.

Windows which have contents exceeding a size of 800 x 600 pixels, are configured with a maximum display size of 800 x 600 pixels in the Title tags. These properties can be changed as needed. This is done so that the window doesn't open so large that the window has to be moved in order to resize it.

For the Assets_v8 window shown above, the (1) Title tag has been set to a maximum size of 800 x 600 pixels, but the size of the (2) canvas object is actually 1251 x 1042. The Canvas object is used in the generation of the XAML windows in order to accurately represent each object's size and position as it was located within the original source database Layout or Form.

.Net Conversion Service - Manual Tasks

Create ADO.NET Data Source

Prior to Building or running the newly generated Visual Studio projects, it is necessary to create a Data Source for the database server which will be used by the project.

.Net Conversion Service Properties - Data Source Name

The screenshot shows the ".Net Conversion Service" window with the "Data Source Name" tab selected. The window has a blue title bar and standard Windows window controls. On the left, there is a sidebar with an icon of a database and a green arrow, and a list of instructions. The main area features a diagram of a database cylinder pointing to a ".NET Framework" box. Below this, there are several input fields and a "Migrate" button. A red rectangle highlights the "Data Source" field, which contains the text "AssetsDB".

Instructions:

- Migrate Databases to .Net Projects:**
- Step 1) Import information about the FileMaker Pro, Access database, or Visual FoxPro project into FmPro Migrator Developer Edition.
- Step 2) Enter the name of the new Visual Studio project which will be created.
- Step 3) Select the project directory where the C# and VB Visual Studio projects will be created.
- Note:** Two new project directories will be created within the selected project


Layout/Form Qty: 2

Project Name: Assets002


Project Directory: C:/ds/vs2010 **Browse**

Data Source: AssetsDB

Processing Type: Licensed

License Key: 

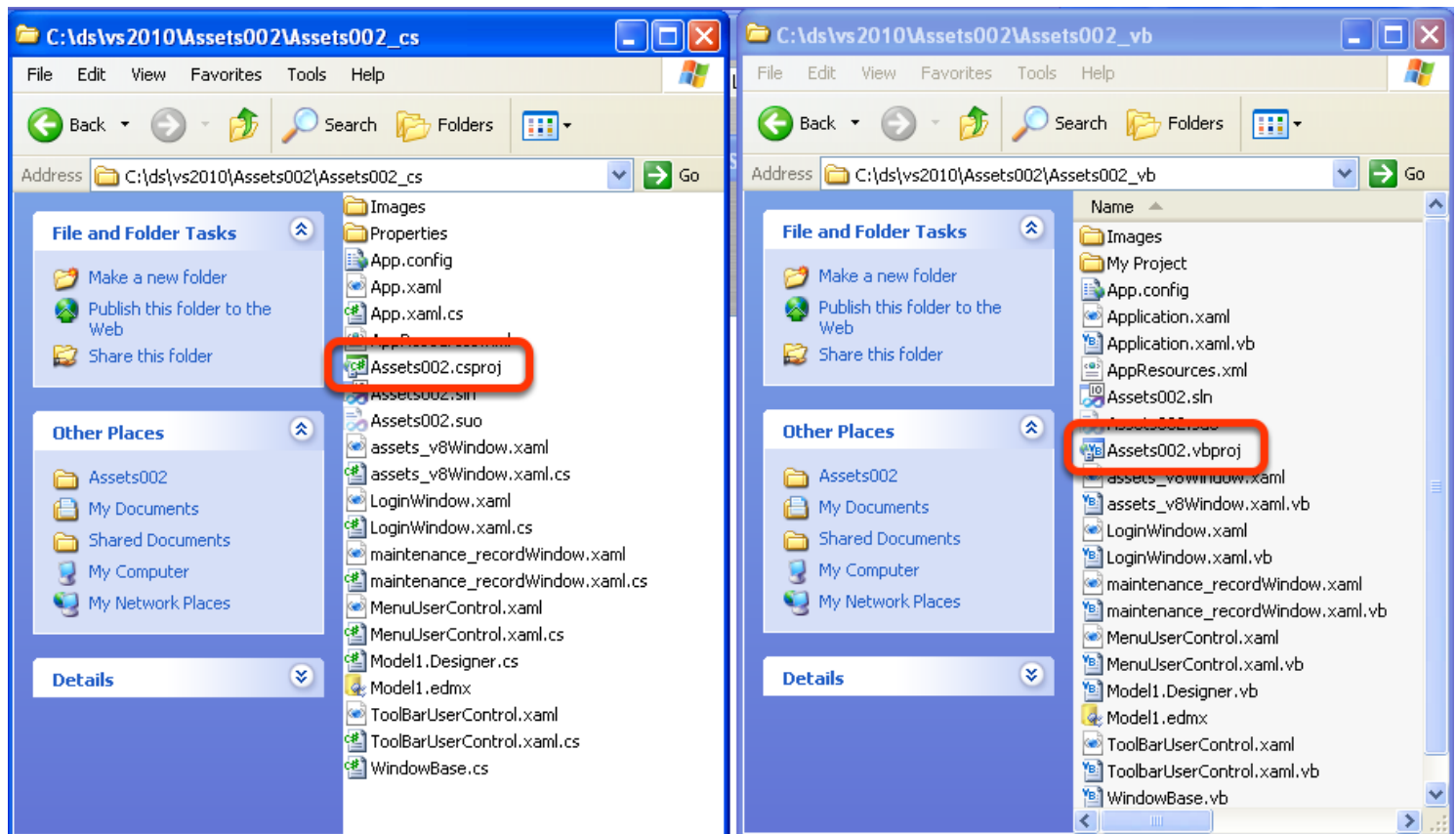
Valid License Key
Valid Until 1/15/2011 (87 Forms)

QUOTE 

Migrate

The screenshots in the ADO.NET Data Source setup use a project named Assets002, using an ADO.NET Data Source named AssetsDB.

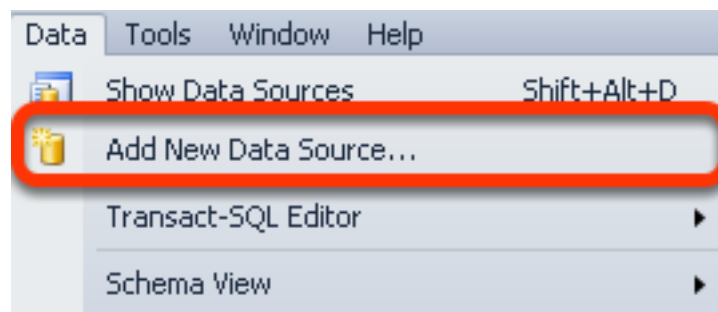
Open Visual Studio Project File



Double-click on either of the Visual Studio project or solution files, to open the project in Visual Studio 2010.

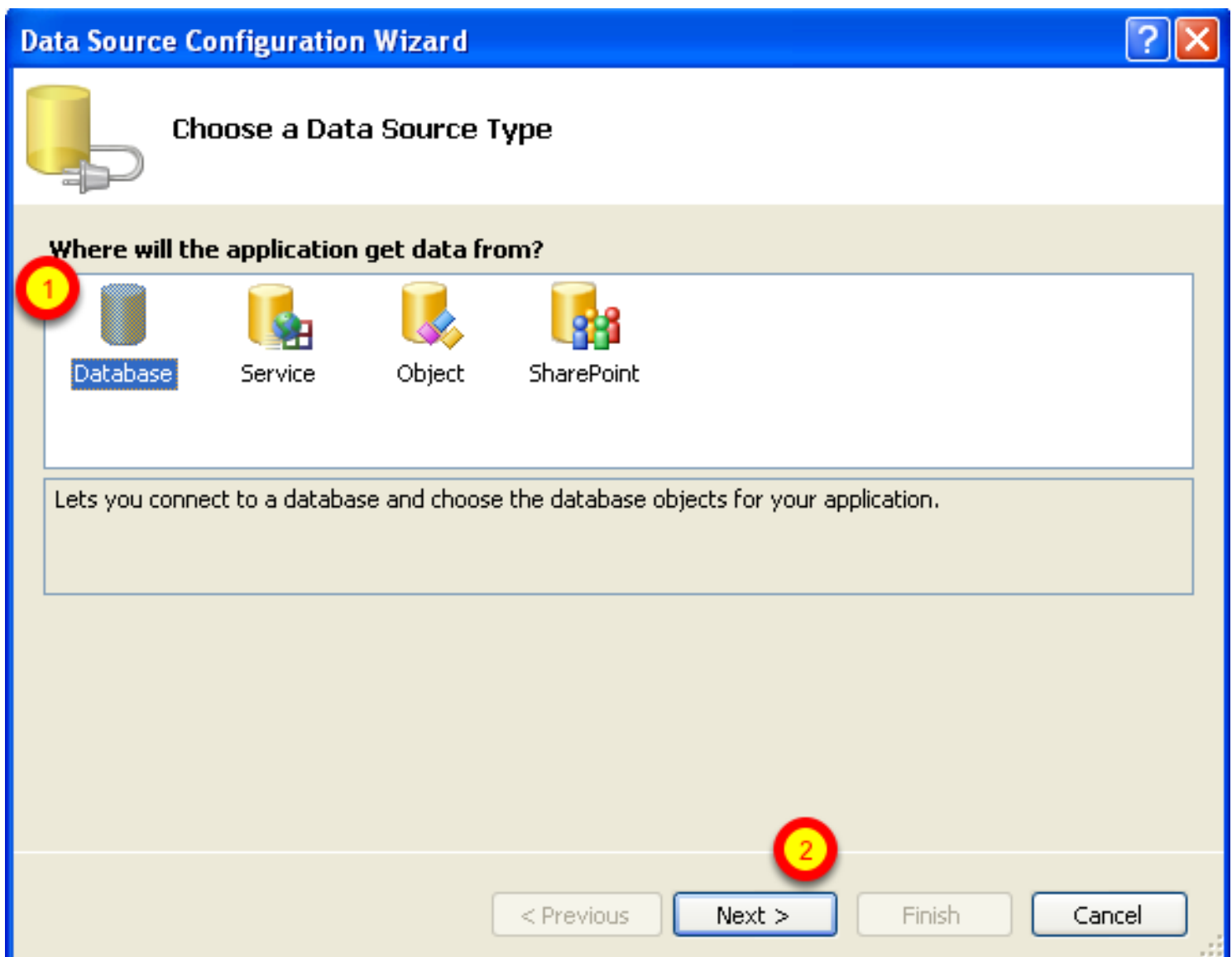
Once the project has been opened, there will be many errors displayed in the Errors list below the development window. These errors are due to the lack of a Data Source for the project, even though the errors may point to src:WindowBase, MenuUserControl or other objects within the project.

Add New Data Source



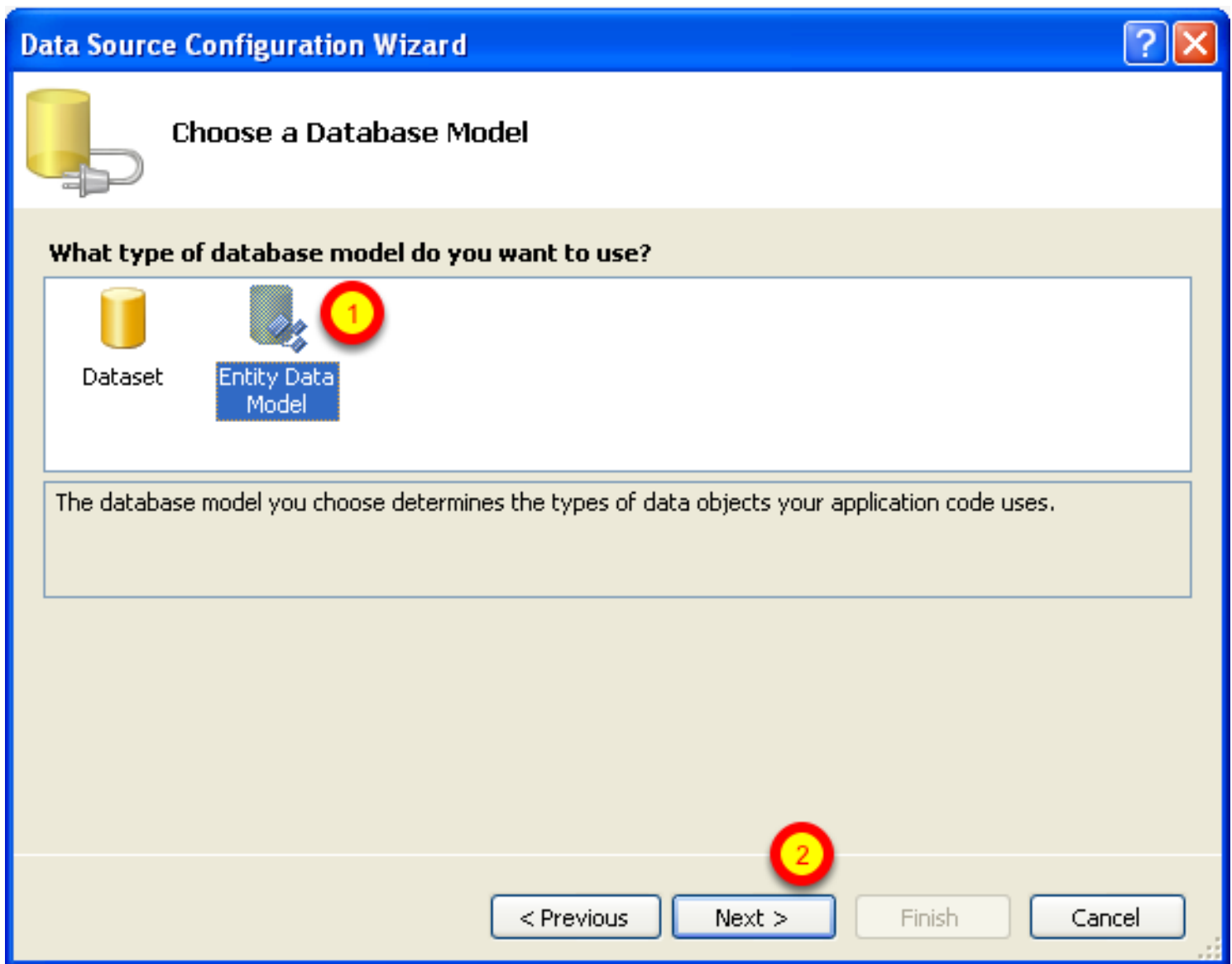
Add a new Data Source from the Data menu.

Choose Database



(1) Click Database, (2) click the Next button.

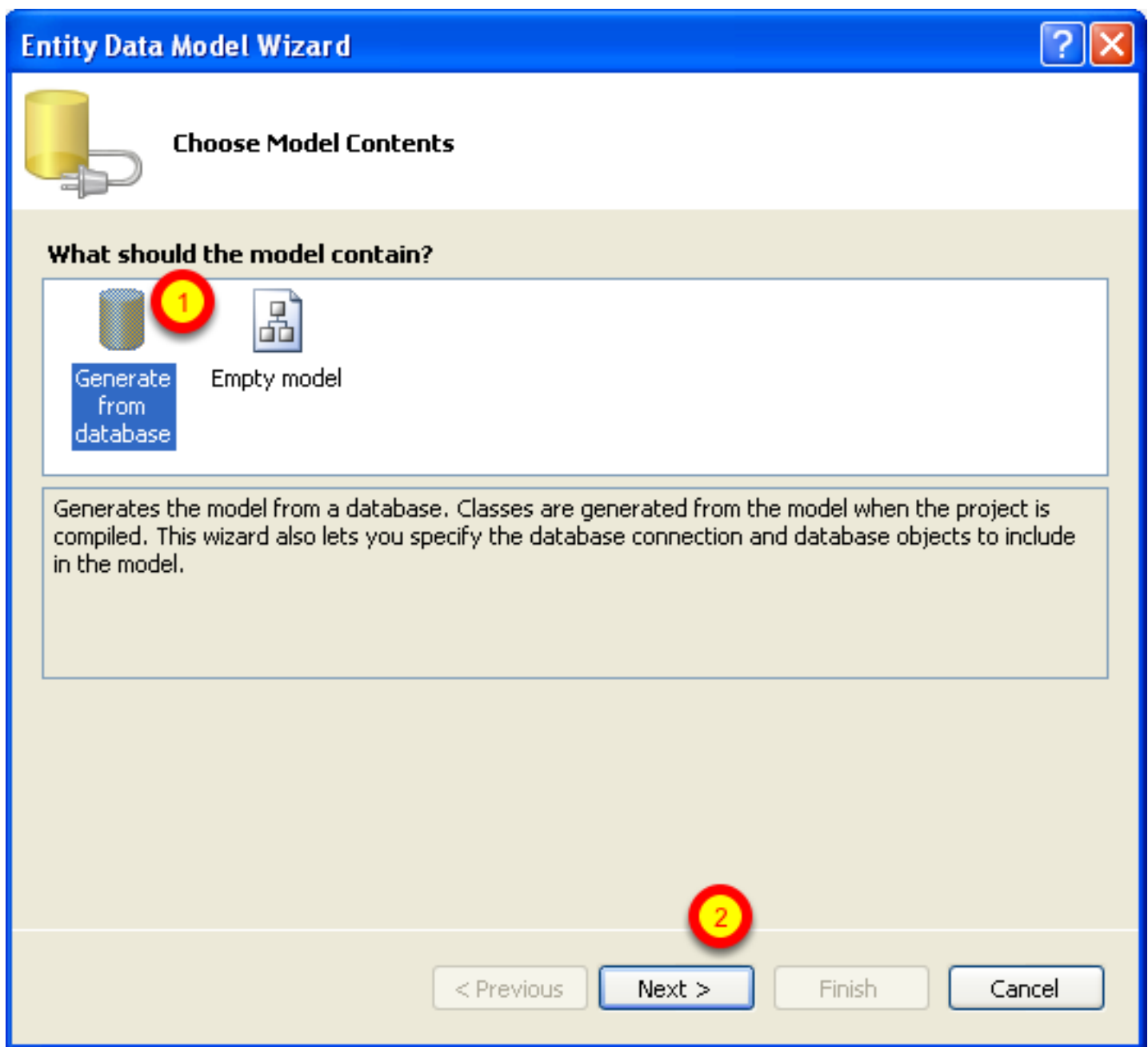
Choose Entity Data Model



(1) Click Entity Data Model, (2) click the Next button.

FmPro Migrator generates code which uses the Entity Data Model, in order to isolate project code from changes within the database.

Choose Generate from database

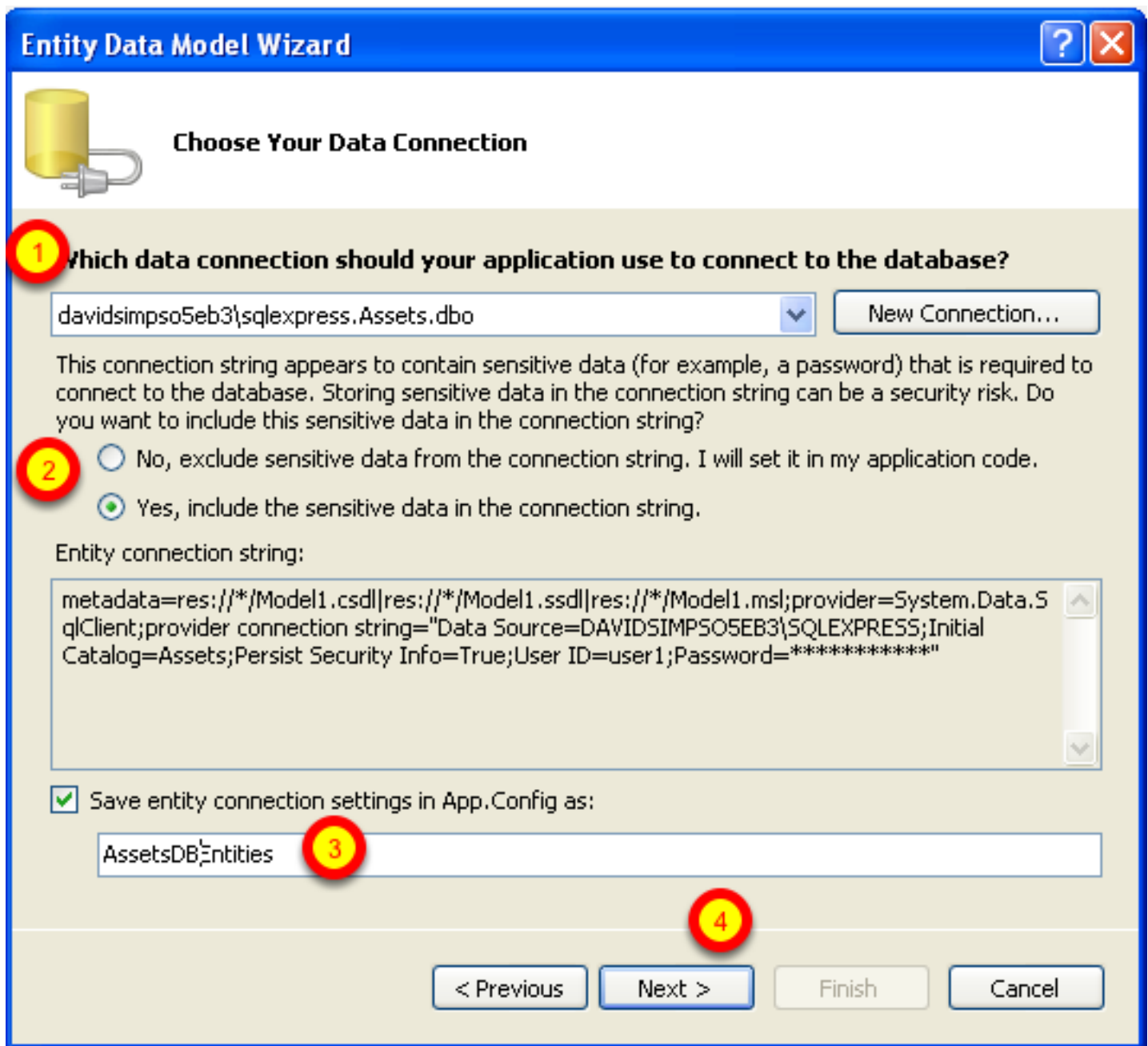


(1) Click Generate from database, (2) click the Next button.

At this point in the conversion process, all of the data and relationships should have been migrated into the SQL database server. This way Visual Studio can gather the relationships directly from the database as it builds the data model.

Note: Some relationships cannot be extracted from all databases by FmPro Migrator. See the manual completion tasks step the discussions concerning the Missing Relationships Report.xls file for more details.

Select Connection, Enter Entity Connection Name



The image shows the 'Entity Data Model Wizard' dialog box. It has a blue title bar with a question mark and a close button. Below the title bar is a yellow cylinder icon with a plug and the text 'Choose Your Data Connection'. The main area is light yellow. It contains a question 'Which data connection should your application use to connect to the database?' with a dropdown menu showing 'davidsimpso5eb3\sqlexpress.Assets.dbo' and a 'New Connection...' button. Below this is a warning message about sensitive data in the connection string. There are two radio buttons: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.' The 'Yes' option is selected. Below the radio buttons is a text box labeled 'Entity connection string:' containing a long string of metadata and connection details. At the bottom, there is a checkbox 'Save entity connection settings in App.Config as:' which is checked. Below the checkbox is a text box containing 'AssetsDBEntities'. At the very bottom are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'. Red circles with numbers 1 through 4 are overlaid on the image to indicate the steps: 1 points to the dropdown menu, 2 points to the radio buttons, 3 points to the 'Save entity connection settings' checkbox, and 4 points to the 'Next >' button.

Entity Data Model Wizard

Choose Your Data Connection

1 Which data connection should your application use to connect to the database?

davidsimpso5eb3\sqlexpress.Assets.dbo New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

2 ☐ No, exclude sensitive data from the connection string. I will set it in my application code.
☒ Yes, include the sensitive data in the connection string.

Entity connection string:

metadata=res://*/Model1.csdl|res://*/Model1.ssdl|res://*/Model1.msl;provider=System.Data.SqlClient;provider connection string="Data Source=DAVIDSIMPSO5EB3\SQLEXPRESS;Initial Catalog=Assets;Persist Security Info=True;User ID=user1;Password=*****"

☒ Save entity connection settings in App.Config as:

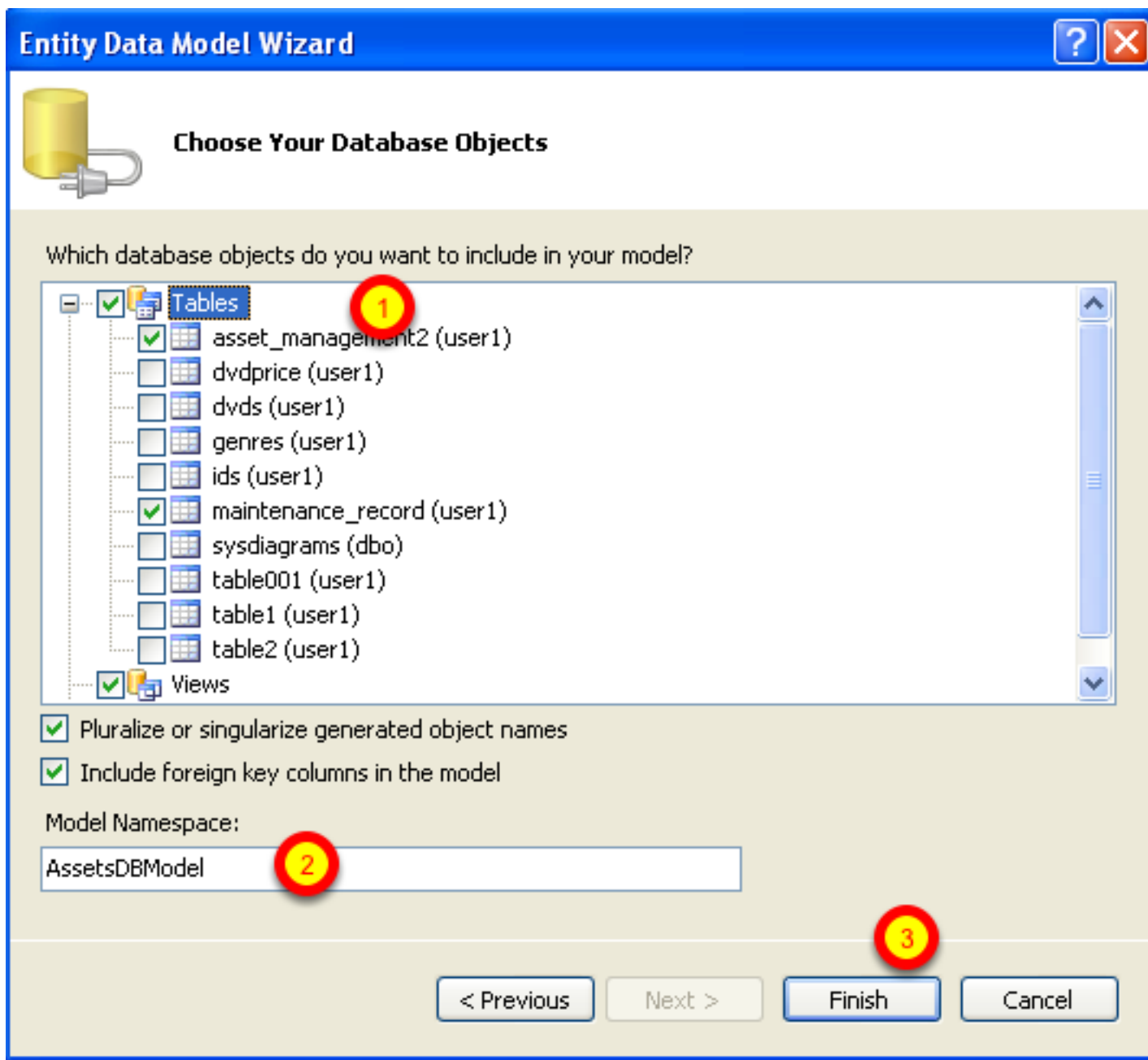
AssetsDBEntities

4

< Previous Next > Finish Cancel

(1) Select an existing data connection name or create a new one, (2) Include or Exclude sensitive data from the connection string, (3) check Save entity connection settings in App.Config file and enter the connection name. In this example, the Data Source name entered into the .Net Conversion Service window was AssetsDB. Therefore the Entity Connection name should be <Data Source Name>Entities - which will become AssetsDBEntities. The Entity Name listed here must exactly match the generated files, or errors will still be displayed in Visual Studio. (4) Click the Next button.

Select Tables, Name the Model



The image shows the 'Entity Data Model Wizard' window, specifically the 'Choose Your Database Objects' step. The window has a blue title bar with the text 'Entity Data Model Wizard' and standard Windows window controls. Below the title bar is a yellow database cylinder icon and the text 'Choose Your Database Objects'. The main area contains the question 'Which database objects do you want to include in your model?'. There are two tree views: 'Tables' and 'Views'. The 'Tables' tree is expanded, showing a list of tables with checkboxes next to them. The 'Views' tree is also expanded, showing a list of views. Below the tree views are two checked options: 'Pluralize or singularize generated object names' and 'Include foreign key columns in the model'. Below these is a text box labeled 'Model Namespace:' containing the text 'AssetsDBModel'. At the bottom of the window are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'. Red circles with numbers 1, 2, and 3 are overlaid on the image. Circle 1 is over the 'Tables' tree view. Circle 2 is over the 'Model Namespace:' text box. Circle 3 is over the 'Finish' button.

Entity Data Model Wizard

Choose Your Database Objects

Which database objects do you want to include in your model?

Tables

- ☒ asset_management2 (user1)
- ☐ dvdprice (user1)
- ☐ dvds (user1)
- ☐ genres (user1)
- ☐ ids (user1)
- ☒ maintenance_record (user1)
- ☐ sysdiagrams (dbo)
- ☐ table001 (user1)
- ☐ table1 (user1)
- ☐ table2 (user1)

Views

☒ Pluralize or singularize generated object names

☒ Include foreign key columns in the model

Model Namespace:

AssetsDBModel

< Previous Next > Finish Cancel

(1) Select the tables, views which should be included in the data model. (2) Enter the Model Namespace. In this example, the Data Source name entered into the .Net Conversion Service window was AssetsDB. Therefore the Model Namespace must be <Data Source Name>Model - which will become AssetsDBModel. The Model Namespace listed here must exactly match the generated files, or errors will still be displayed in Visual Studio. (4) Click the Finish button.

Re-Generating The Project

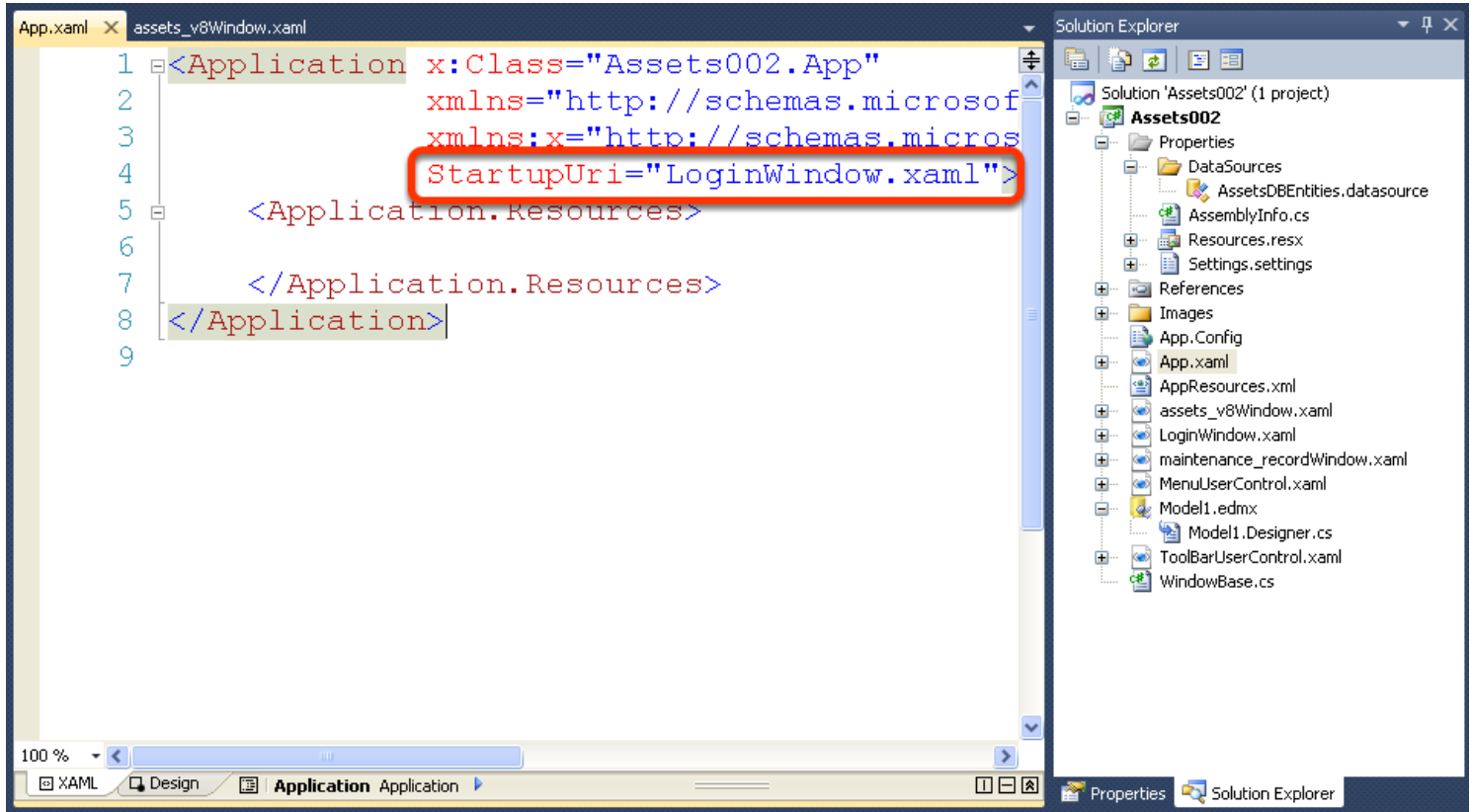
FmPro Migrator avoids writing over the Visual Studio project files containing the Data Source information. Therefore if the project needs to be re-created, the same Data Source will be reused.

From a troubleshooting standpoint, this is significant because in some cases you might want to start from the beginning. If this is the case, Exit Visual Studio and delete the top-level project directory. However, if the Visual Studio Debugger has been used on a project, it seems that it is usually necessary to manually delete the contents of the bin folder which containing the <Project Name>.vhost.exe file.

Manual Tasks - .Net Conversion Service

There are some processing steps which must be completed manually when performing a .Net Conversion.

Set StartupUri



By default, each project will open the LoginWindow.xaml window. The App.xaml file can be changed to select another form in the project.

Also, any application specific security code would need to be added to a new LoginWindow.cs/vb file in order to avoid having the data source connection information exposed within the App.Config file.

Layout Parts - Header, Footer, Subsummary

The implementation of Report Header, Footer, Subsummary parts requires application-specific development code within code behind files.

Unsupported Form/Report Objects

Some Layout/Form objects are not supported for automated conversion as listed below:

FileMaker Pro Layouts:

Merge Fields

Layout Date, Time, Page Number objects

Chart Objects

WebView

Microsoft Access Forms/Reports:

acCustomControl (ActiveX)

chart

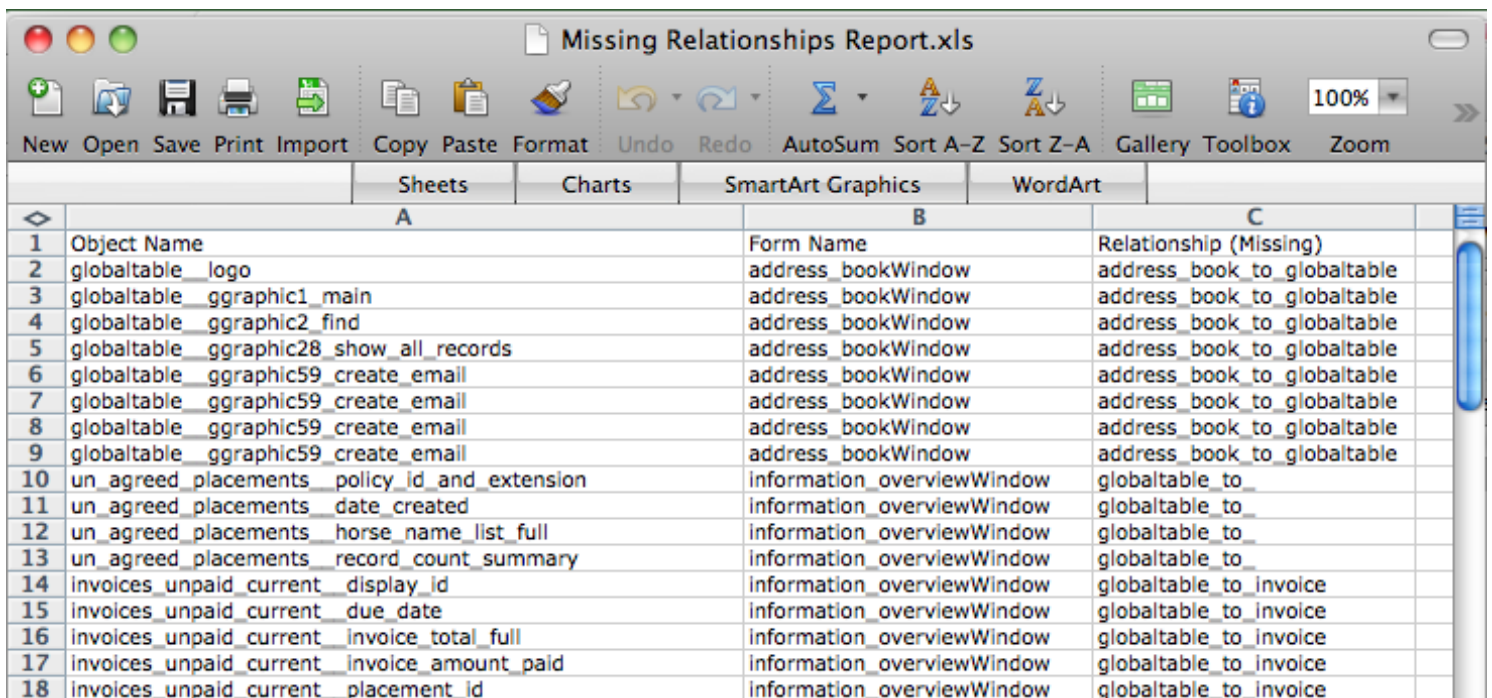
Visual FoxPro Forms/Reports:

Timer

OLEControl

HyperLink

Missing Relationships Report.xls



	Sheets	Charts	SmartArt Graphics	WordArt
	A	B	C	
1	Object Name	Form Name	Relationship (Missing)	
2	globaltable_logo	address_bookWindow	address_book_to_globaltable	
3	globaltable_ggraphic1_main	address_bookWindow	address_book_to_globaltable	
4	globaltable_ggraphic2_find	address_bookWindow	address_book_to_globaltable	
5	globaltable_ggraphic28_show_all_records	address_bookWindow	address_book_to_globaltable	
6	globaltable_ggraphic59_create_email	address_bookWindow	address_book_to_globaltable	
7	globaltable_ggraphic59_create_email	address_bookWindow	address_book_to_globaltable	
8	globaltable_ggraphic59_create_email	address_bookWindow	address_book_to_globaltable	
9	globaltable_ggraphic59_create_email	address_bookWindow	address_book_to_globaltable	
10	un_agreed_placements_policy_id_and_extension	information_overviewWindow	globaltable_to_	
11	un_agreed_placements_date_created	information_overviewWindow	globaltable_to_	
12	un_agreed_placements_horse_name_list_full	information_overviewWindow	globaltable_to_	
13	un_agreed_placements_record_count_summary	information_overviewWindow	globaltable_to_	
14	invoices_unpaid_current_display_id	information_overviewWindow	globaltable_to_invoice	
15	invoices_unpaid_current_due_date	information_overviewWindow	globaltable_to_invoice	
16	invoices_unpaid_current_invoice_total_full	information_overviewWindow	globaltable_to_invoice	
17	invoices_unpaid_current_invoice_amount_paid	information_overviewWindow	globaltable_to_invoice	
18	invoices_unpaid_current_placement_id	information_overviewWindow	globaltable_to_invoice	

It is not always possible to export all of the relationships from a particular database for use in the automated conversion process. Within FileMaker Pro databases, some relationships are inferred automatically from existing relationships. Within Microsoft Access and Visual FoxPro projects, relationships may be implemented via Queries or application-specific programming code. Therefore FmPro Migrator analyzes the project as the objects are being created within each XAML file. If a relationship cannot be found for a field in another table, this information will be written into the Missing Relationships Report.xls file.

The columns in the report show:

Object Name - The name of the object as it has been created in the XAML file.

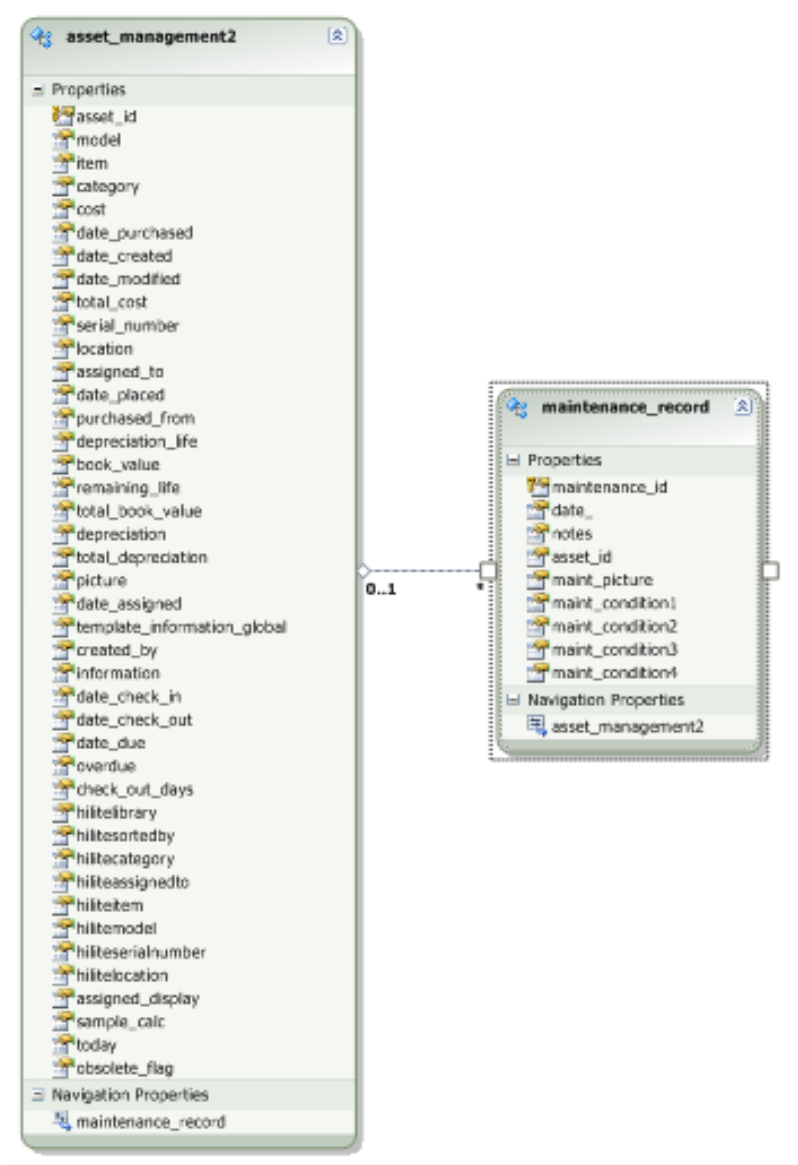
Form Name - The name of the form.

Relationship (Missing) - The Left Table and Right Table names for the proposed relationship to be created.

Within this file, the same missing relationship will be listed multiple times - as it will be listed for each object. Therefore fixing the problem for one field will likely fix the problem for many fields. Some relationship names will appear empty or incomplete - if there wasn't enough info available to determine the table names required for the relationship.

This report is written into the FmPro Migrator output directory.

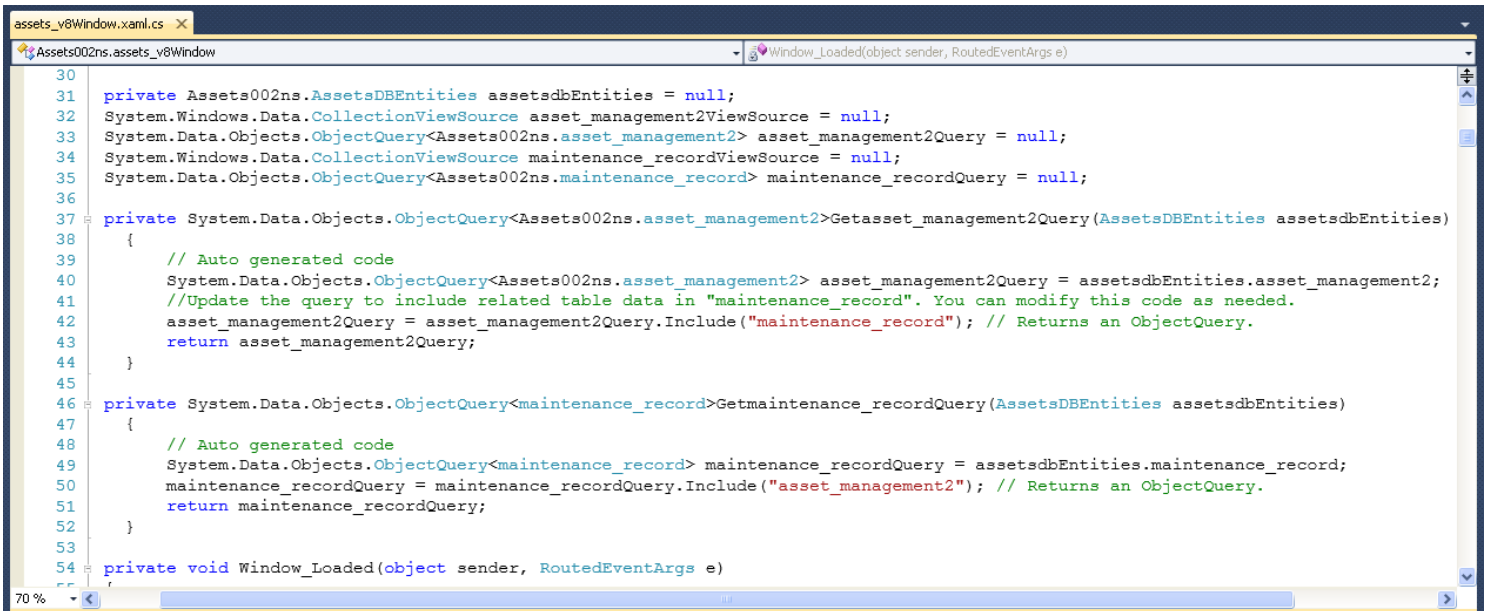
Entity Model Designer



Visual Studio creates the data model and entity framework code from the database connection. If changes are made to the database, the model can be updated from the updated database schema. Or changes can be made to the model without making changes to the database. Therefore either

technique can be used to resolve issues with missing relationships.

.Net Relationship Info.cs



```
30
31 private Assets002ns.AssetsDBEntities assetsdbEntities = null;
32 System.Windows.Data.CollectionViewSource asset_management2ViewSource = null;
33 System.Data.Objects.ObjectQuery<Assets002ns.asset_management2> asset_management2Query = null;
34 System.Windows.Data.CollectionViewSource maintenance_recordViewSource = null;
35 System.Data.Objects.ObjectQuery<Assets002ns.maintenance_record> maintenance_recordQuery = null;
36
37 private System.Data.Objects.ObjectQuery<Assets002ns.asset_management2> Getasset_management2Query(AssetsDBEntities assetsdbEntities)
38 {
39     // Auto generated code
40     System.Data.Objects.ObjectQuery<Assets002ns.asset_management2> asset_management2Query = assetsdbEntities.asset_management2;
41     //Update the query to include related table data in "maintenance_record". You can modify this code as needed.
42     asset_management2Query = asset_management2Query.Include("maintenance_record"); // Returns an ObjectQuery.
43     return asset_management2Query;
44 }
45
46 private System.Data.Objects.ObjectQuery<maintenance_record> Getmaintenance_recordQuery(AssetsDBEntities assetsdbEntities)
47 {
48     // Auto generated code
49     System.Data.Objects.ObjectQuery<maintenance_record> maintenance_recordQuery = assetsdbEntities.maintenance_record;
50     maintenance_recordQuery = maintenance_recordQuery.Include("asset_management2"); // Returns an ObjectQuery.
51     return maintenance_recordQuery;
52 }
53
54 private void Window_Loaded(object sender, RoutedEventArgs e)
```

Each code behind file contains ObjectQuery code intended to be used for accessing fields within the table associated with the Layout/Form and also with related database tables. This code is built during the analysis and creation of objects on each window. If the relationship structure changes too much during the process of adding missing relationships to the data model, then this code may also need to be modified. But by default, the code is designed with the assumption that all of the fields on the window are either associated with the 1st table listed or are located in a related table.

Grid Column Sizes

FmPro Migrator automatically creates a new portal to represent any SubForms/SubReports found on the original Access Form/Report. Unlike FileMaker portal objects, Access SubForm/SubReport objects contain horizontal scrollbars. Therefore there could be too many fields incorporated on the original SubForm/SubReport to make a readable display within the area of the new FileMaker portal. FmPro Migrator

It isn't possible to determine the width of columns located on Microsoft Access SubForms or Visual FoxPro Grid objects. Therefore FmPro Migrator divides the width of the portal by the number of fields and creates each field of the same size. The fields are also created in the same left to right display order as they were created on the original SubForm/Grid. It may be necessary to manually resize the individual WPF Grid fields to accommodate the actual data which will be displayed within the Grid.

Code Conversion

All of the code from the original database is copied over to the new .Net project as commented code within the App.xaml.cs/Application.xaml.vb files. This code is intended to serve as a guide when writing new .Net application code having the same functionality.

Overlapping Layout Objects

There isn't any method to determine the top to bottom stacking order of objects on an Access Form/Report. FmPro Migrator builds objects within the new XAML file in a specific logical order to minimize issues with overlapping objects. The order of creating layout objects is:

Rectangle

Line

Graphic Image

Text Label

Field

Some cosmetic changes will potentially still need to be performed manually to fine tune the stacking order.

Duplicate Objects Report.xls

Object Type	Object Name	Notes
Layout	Horse_New	Skipped - Reason: Duplicate Layout Contents.
Layout	Horse_Cancel	Skipped - Reason: Duplicate Layout Contents.
Layout	Horse_Cancel_Confirm	Skipped - Reason: Duplicate Layout Contents.
Layout	Placement Form - Print	Skipped - Reason: Duplicate Layout Contents.
Layout	Find All Layout2 (Claim)	Skipped - Reason: Duplicate Layout Contents.
Layout	Address Labels	Skipped - Reason: Duplicate Layout Contents.

Value Lists having exactly the same name and contents and Layouts having exactly the same contents will be skipped during the conversion process, and will be reported in the Duplicate Objects Report.xls file. During the capture of layouts from FileMaker Pro, it is possible that a file could accidentally be captured twice. Before making manual modifications, to the XAML, vb/cs files - this report should be checked. If there were no objects skipped during processing, a report will not be created.

This report is written into the FmPro Migrator output directory.

Missing Layout Table Report.txt

```
Total Layouts = 87
Layouts Skipped = 9
Layout Name TO Name

_Temp_Endorsement_Line_Item _Temp_Endorsement_Line_Item
_Temp_Policy_Renewal_Horses _Temp_Policy_Renewal_Horses
Renewal Letter - Print _Temp_Policy_Renewal_Horses
Loss Ratio Report - UW Contract _Temp_Loss_Ratio_Report
Loss Ratio Report - Underwriter _Temp_Loss_Ratio_Report
Binder- new business
Policy Subjectivity
Envelope
```

FmPro Migrator attempts to associate each Layout/Form with a database table. This information is used to create the CollectionViewSource in the XAML and code behind files. If a table is not associated with a Layout/Form, its name will be listed in the Missing Table Report.txt. This report is written into the FmPro Migrator output directory. Fixing this issue will require manual intervention.

- 1) Manually associate a table with each Layout in on the Layouts tab of the FmPro Migrator Migration Process window.
- 2) Or change the XAML and code behind files appropriately so that the CollectionViewSource and query info is not being used. This situation might occur if you are manually writing your own code to populate the objects in the window.