A Generic Import Utility, Part 2

Doug Hennig

Part 1 of this two-part series presented a set of classes making up a generic import utility you can add to your applications to provide import capabilities from a variety of data sources. Part 2 focuses on the user interface components.

As we saw in the last issue of FoxRockx, the classes in SFImport.VCX provide the engine components for a generic import utility. The classes in SFImportUI.VCX provide the user interface components. The main class, SFImportWizard, provides a wizard-based dialog allowing the user to select the file to import from (the source), the table to import into (the target), map fields from the source to the target, and perform the import. Rather than examining the code in this and the other class in SFImportUI.VCX, I'll discuss how the dialog works from a user perspective plus point out some places you may wish to customize. Feel free to examine the code if you want to know how it works.

To run the wizard, instantiate SFImportWizard and pass it the name of the database the tables to import into are in; since the Import Wizard has a private datasession, it cannot see the current database. TestWizard.PRG, included with this issue's downloads, is a sample program that instantiates the wizard.

```
open database TESTDATA
loForm = newobject('SFImportWizard', ;
  'SFImportUI.vcx', '', 'TESTDATA')
loForm.Show(1)
```

Although VFP supports many file types with the IMPORT FROM command, I decided to use only the more common ones in the Import Wizard: comma-separated values (CSV); flat text files, also known as "simple data format" or SDF; DBF files; and Microsoft Excel documents. See the **Customizing the Import Wizard** section later in this article if you want to provide more file types.

The Import Wizard has four or five steps, depending on the type of file you import from. The first step, shown in **Figure 1**, is selecting the file to import from. If you select *Import a new file*, type the name of the import file in the *Import file* text box or click the button beside the text box and select the file from the Open File dialog that appears. The Import Wizard identifies the type of file you select and displays the type in the *File type* combo box. You can change the file type if it misidentified the file. It also tries to determine whether a header record exists for CSV files and sets *File has header record* accordingly. You can, of course, change that if necessary. Try importing from the sample files CustCSVH.TXT, which has a header record, and CustCSV.TXT, which does not, to see the difference.

If you select *Import a new file using an existing profile* instead, you can then select a profile from a list of previously saved profiles. Profiles, discussed in the last issue, save time in defining how to import from a specific type of file. For example, if you receive an import file with the same structure every month, you only have to specify the layout of the file in the Import Wizard the first time. You can then save the layout as a profile, and the next time you need to import from that file type, select the saved profile. The Delete button removes the selected profile from the list.

Profiles are saved in tables in the Profiles subdirectory of the current folder. Profiles.DBF is a simple table containing just two columns: the descriptive name of the profile (the one shown in the list box in Step 1) and the name of the DBF file that contains the profile settings.

Selecting a profile fills the settings in the Import Wizard with the values saved in the profile, including the name of the file to import from. You can change the file name if necessary, but the field mappings, discussed later, don't change, so be sure to select a file with a structure matching the profile.

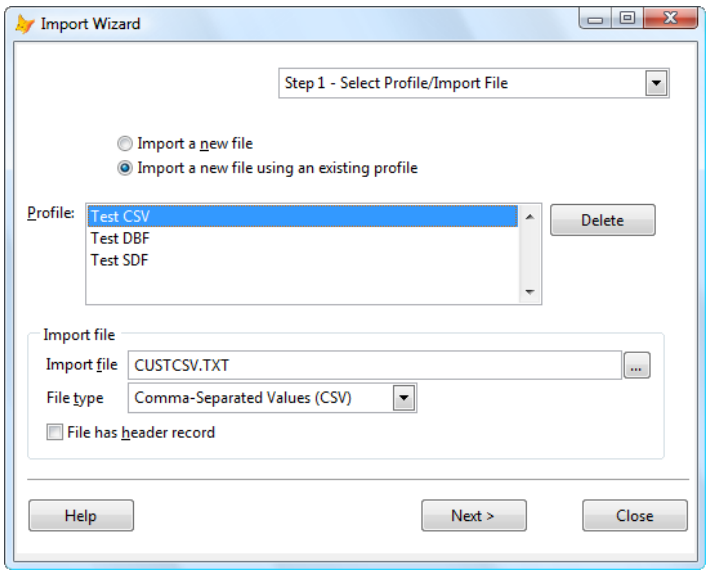

Figure 1. In Step 1, select either a saved profile or the file to import from.

Click Next to move to the next step. You can also select the desired step from the step combo box at the top of the dialog.

Step 2, Define File Structure, only appears when you import from a CSV or SDF file, and how this step appears depends on which of those two types you select. **Figure 2** shows the step for CSV files. The grid displays the name of each field in the import file and the contents of the first record. If the CSV file has a header record, it reads the field names from that record. If not, it assigns names like "FIELD001" and "FIELD002" to the fields. To change the name of the field selected in the grid, click the Name button or double-click the field and enter the desired field name in the dialog that appears. A field must be named using the VFP field name convention; that is, it must start with a letter and can contain only letters, numbers, and underscores. The Import Wizard reads the first ten records from the import file into a cursor, and you can click the Previous and Next buttons to move through these records to see what the file looks like.

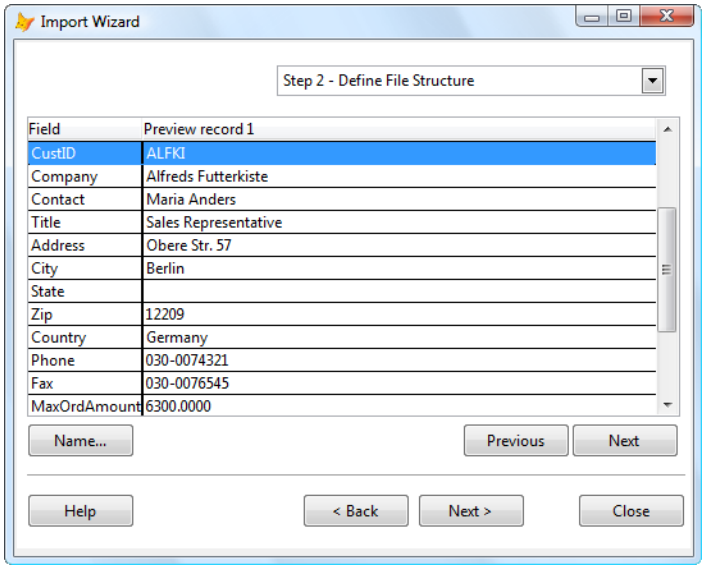

Figure 2. For CSV files, Step 2 shows the contents of the file and allows you to name the fields if the file doesn't contain a header record.

SDF files are more work to use because they don't have a defined structure like CSV files do. Instead, each field is a fixed length. SDF aren't that common in the PC world but they are often used as an export format from mainframe systems. Here's an example:

```
ALFKIAlfreds Futterkiste     Maria Anders
ANATRAna Trujillo EmparedadosAna Trujillo
```

As a result, Step 2 is more complex for SDF files because you have to tell the Import Wizard where each field ends. **Figure 3** shows an example of the structure of the sample file CustSDF.TXT.

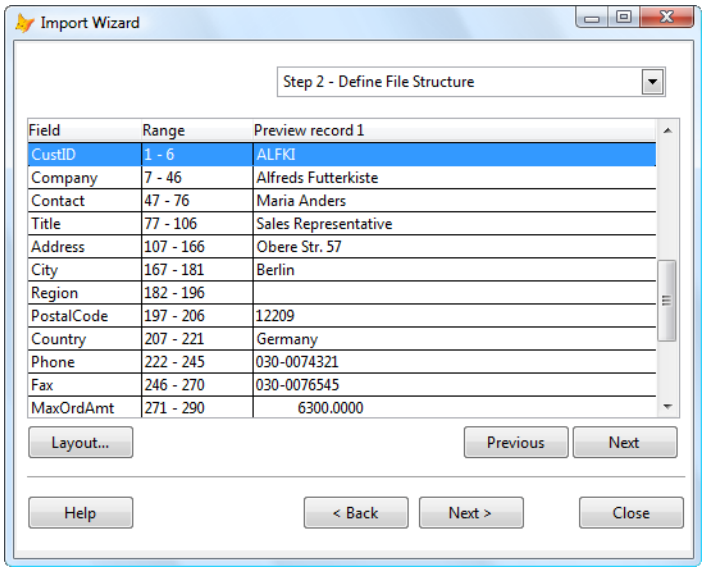

Figure 3. For SDF files, Step 2 allows you to define the layout of the file.

Initially, the Import Wizard shows a single field, named FIELD001, with a range of 1 to the number of characters per line of text. Double-click the field or click the Layout button to display the dialog shown in **Figure 4**. Enter the desired field name and adjust the starting and ending positions for the field. The controls in the bottom half of the dialog show the contents of the first several lines of the file, a ruler showing the text position, and a red rectangle outlining the field based on the starting and ending position. This makes it easy to see that you've specified the correct values. When you click OK to save the field, the Import Wizard adjusts the range for the field in the grid and automatically creates a new field taking up the rest of the line length.

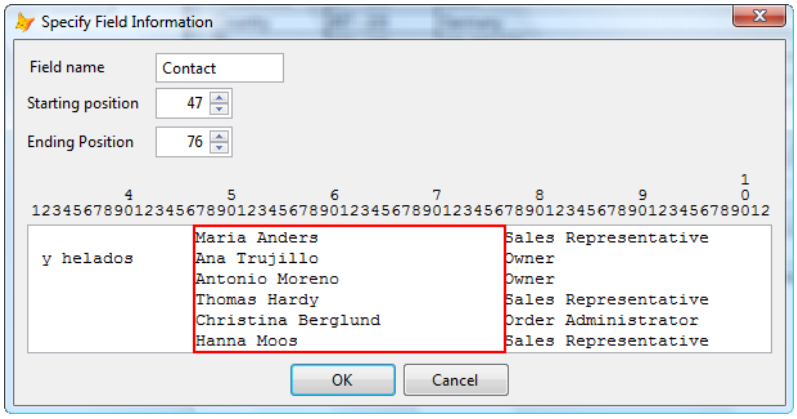

Figure 4. Clicking on Layout in Step 2 displays this dialog, in which you can specify the name, starting, and ending positions of the selected field.

For example, CustSDF.TXT initially has one field named FIELD001 with a range of 1 – 290. Double-click the field, enter CustID as the field name, 1 as the starting position, and 6 as the ending position, and click OK. The grid now shows the CustID field, its range and contents for the first record, and a new Field2 with a range of 7 – 290. Continue editing the field and adding new ones until you've accounted for the entire structure of the file.

In Step 3 (Step 2 for anything but CSV and SDF files), select the table to import from (see **Figure 5**). The combo box shows all tables in the specified database, but you can tell it to exclude a table by adding "*:SFIMPORT NOIMPORT" to the table's Comment property in the database.

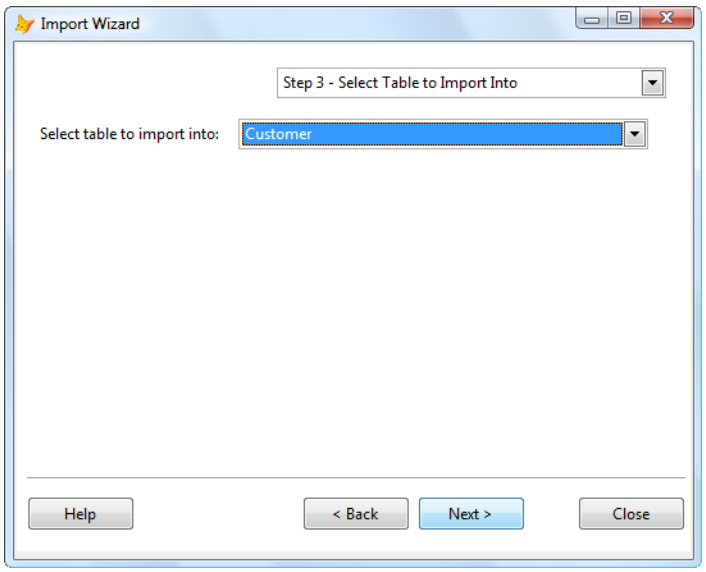

Figure 5. Select which table to import into in Step 3.

The next step, shown in **Figure 6**, allows you to specify how the import file fields map to the selected table's fields. The Import Wizard automatically maps fields with identical names, so some of the mapping may already be done for you. To map a pair of fields, click the two fields in the *Import Fields* and *Import Into* lists and click the Map button. The name of the import field appears in the Expression column in the *Import Into* list, indicated the mapping. If you want to use an expression rather than a field, click the Expression button and enter a valid VFP expression. For example, Figure 6 shows that the import PostalCode field is upper-cased when imported into the table's Postal Code field. The Import Wizard automatically uses a CAST() expression if you map two fields with different data types. For example, the MaxOrdAmt field in CustCSV.TXT contains text (since all fields in a CSV or SDF file are considered text) so the import process uses CAST(MaxOrdAmt as Y) when importing it into the Maximum Order field.

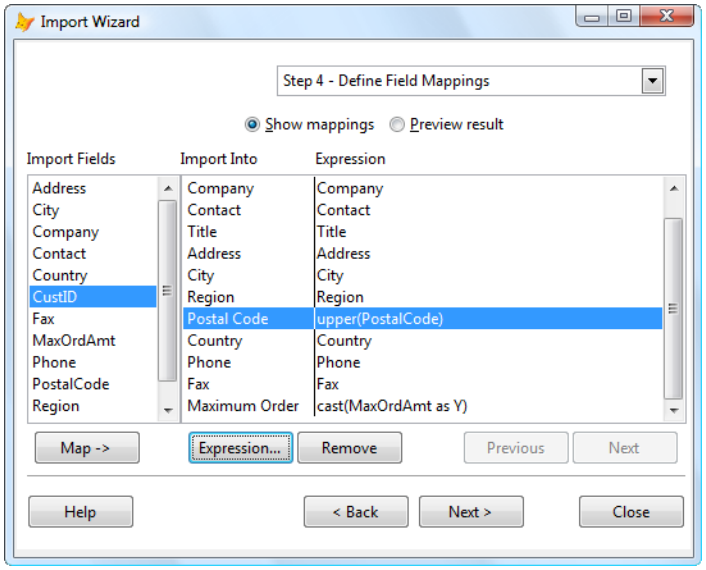

Figure 6. Map each import field to a table field in Step 4. You can specify expressions to perform data conversion.

To clear a mapping, select the field in the *Import Into* list and click the Remove button. To see how the import will work, select *Preview result*. As **Figure 7** shows, the grid displays the results of each mapping expression for the first record. Click the Previous and Next buttons to preview the results for other records.

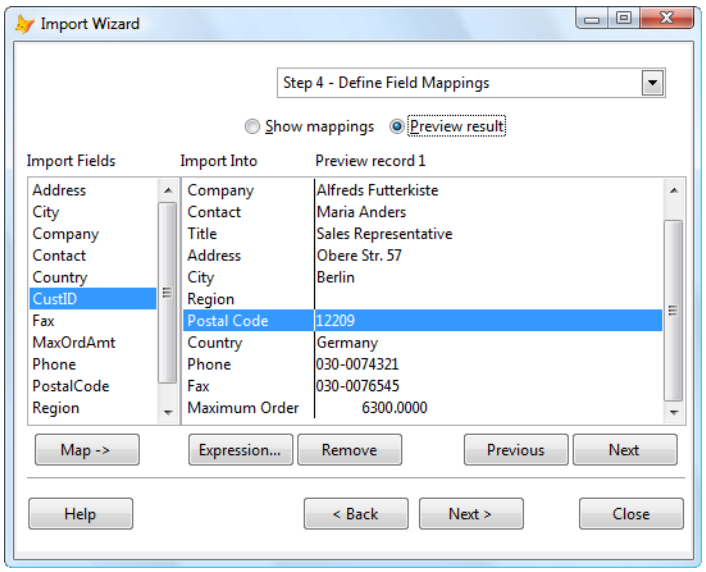

Figure 7. Select Preview Result to see how the field mappings work when the data is imported.

The last step (**Figure 8**) is performing the import. Click the Start button to start the import process. This step displays the progress of the import, including the total number of records in the import file, the number of records processed so far, how many were imported, and how many were rejected because they were duplicates, invalid, or were filtered out. While the import proceeds, a meter indicates the progress and the Start button appears as Cancel. Click this button to stop the import.

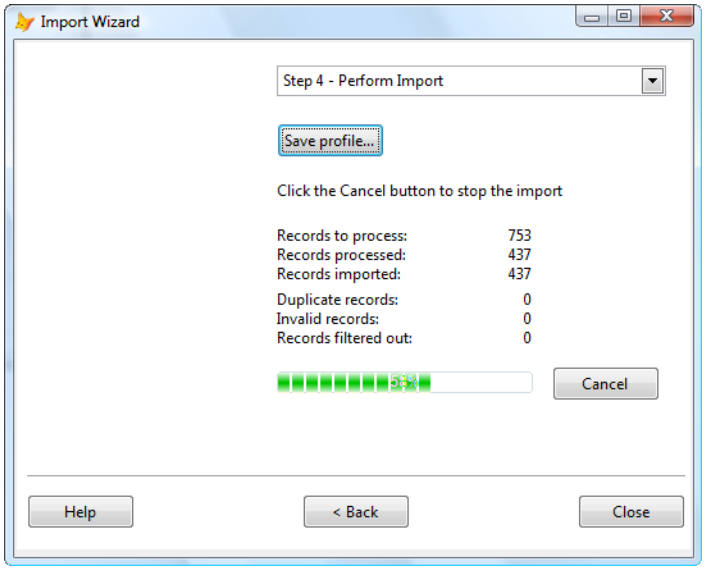

Figure 8. The dialog shows the progress of the import, including how many records were imported and how many rejected.

To save your import settings as a profile, click the Save Profile button and enter the desired name. The Import Wizard creates a profile table with that name in the same directory as Profiles.DBF and adds a record for that profile to the Profiles table so it appears in the wizard the next time you run it.

Trying it out

The download files for this article include several sample import files and a sample database and Customer table you can import into.

- CustCSV.TXT is a sample CSV file without a header record, so the Import Wizard names its fields FIELD001, FIELD002, and so forth.
- CustCSVH.TXT is also a CSV file, but it does have a header record, so the field names come from that record.
- CustSDF.TXT is a sample SDF file.

- CustomerImport.DBF is a VFP table with a different structure than the Customer table to import into, so you can see how field mapping works with DBF files.
- CustImport.XLS is an Excel file.

Customizing the Import Wizard

I designed the Import Wizard to be flexible. You can customize much of its behavior by subclassing SFImportWizard and changing some properties or overriding some methods. Here are some ideas.

- If you want to change the path for the profiles tables, set the cProfilesTable property to a different value.
- The DefineFileTypes method specifies the file types supported. You can add other file types if necessary. See the comments in this method about what the various columns in the aFileTypes array contain.
- Override the GetTables method to control what tables appear in the *Select Table to Import Into* step. The first column of the aTables array contains the caption for the table and the second column contains the table name.
- The dialog shown in Figure 4 comes from the SFImportSDFFieldDialog class in SFImportUI.VCX. If you want to use a different dialog, either a subclass of SFImportSDFFieldDialog or a completely new dialog, set the cSDFClass and cSDFLibrary properties of SFImportWizard to the desired class and library.
- The Import Wizard can remember its size and position from the previous time it was used if you set the cRegistryKey property to the registry key under HKEY_CURRENT_USER to store this information. An example is "Software\Stonefield Software Inc.\My Cool Application\Import Wizard Settings."
- If you want context-sensitive help for the Import Wizard, set cHelpFile to the name of the help file and HelpContextID to the ID for the help topic.
- SFImport doesn't have a filter set but as I discussed in the previous issue, it supports filtering so you can specify a filter in the cFilter property of the oImport object in a subclass of SFImportWizard or create a subclass of SFImport that has its cFilter property set accordingly and tell the Import Wizard to use that subclass by setting its cImportClass and cImportLibrary properties.

Summary

The classes presented in this and the previous issue's article provide a generic import utility you can add to your applications. This utility allows your users to import external data into their tables, which is very useful if they're migrating from another application or need to read in data produced by other programs. It was designed to be modular so you can subclass any of the components to change the behavior as desired. I hope you enjoy using this utility.

Doug Hennig is a partner with Stonefield Systems Group Inc. and Stonefield Software Inc. He is the author of the award-winning Stonefield Database Toolkit (SDT); the award-winning Stonefield Query; the MemberData Editor, Anchor Editor, and CursorAdapter and DataEnvironment builders that come with Microsoft Visual FoxPro; and the My namespace and updated Upsizing Wizard in Sedna. Doug is co-author of the "What's New in Visual FoxPro" series (the latest being "What's New in Nine") and "The Hacker's Guide to Visual FoxPro 7.0." He was the technical editor of "The Hacker's Guide to Visual FoxPro 6.0" and "The Fundamentals." All of these books are from Hentzenwerke Publishing (http://www.hentzenwerke.com). Doug wrote over 100 articles in 10 years for FoxTalk and has written numerous articles in FoxPro Advisor and Advisor Guide. He has spoken at every Microsoft FoxPro Developers Conference (DevCon) since 1997 and at user groups and developer conferences all over the world. He is one of the administrators for the VFPX VFP community extensions Web site (http://www.codeplex.com/VFPX). He has been a Microsoft Most Valuable Professional (MVP) since 1996. Doug was awarded the 2006 FoxPro Community Lifetime Achievement Award (http://fox.wikis.com/wc.dll?Wiki~FoxProCommunityLifetimeAchievementAward~VFP). Web: www.stonefield.com and www.stonefieldquery.com, Email: dhennig@stonefield.com, Blog: http://doughennig.blogspot.com