

Installing DataEase Custom Defined Function Libraries (CDFs)

by Lawrence Fox
ComputerWizard Consulting

Version 2.0 (May 2003)

DataEase contains more than fifty functions which have been built into it by the developers—functions such as “JoinText”, “SpellDate”, and “Random”. But each user application is unique, and has its own special requirements. So DataEase allows the application developer to “plug-in” extra commands as and when they’re needed. These plug-in commands are called Custom Defined Functions.

Your DataEase package contains more than two hundred of these functions, ready for immediate use. Many more CDFs have been produced by third-party developers, and can be purchased from them.

From The DataEase CDF Help File
© Sapphire International

ComputerWizard Consulting is one of those third party developers. This White Paper is intended as a general introduction into the strategies that can be used to install and deploy CDF libraries created by Sapphire, ComputerWizard Consulting or others.

A CDF library is a windows program file (called a dynamic link library), usually with a file extension of DLL. A library can contain one or more functions.

Installing a CDF library and its functions into a DataEase application is multi-step process.

1. Decide on a deployment/installation strategy (as discussed in this paper).
2. Copy the files to a folder on your disk(s) based on the strategy.
3. Register the functions with each application that will use them.

1) **Decide on a deployment/installation strategy**

There is no “install” procedure for CDF libraries as far as Windows is concerned—you need only copy the **.dll** to a folder and you’re done. (Libraries distributed by ComputerWizard Consulting include documentation and a few files that assist in installation).

You have four options in installing your CDF libraries:

- a) Use the default installation (the DataEase program folder).
- b) Keep all the CDFs in their own folder under the DataEase program folder.
- c) Copy all of the CDFs into the application (data) folder.
- d) Combine strategies (a) and (c).

Let's look at these options in greater detail:

a) Use the default installation

A default DataEase install creates a set of folders and subfolders similar to the one in Figure 1.

While you can select the drive and main folder during the DataEase installation procedure, you cannot change the subfolders created before the installation, and something very close to Figure 1 is what you'll see.

There is a folder called **CDFLibs** under which there is one folder for each CDF library, containing source files, help files and the actual CDF program (usually a dynamic link library—a .dll).

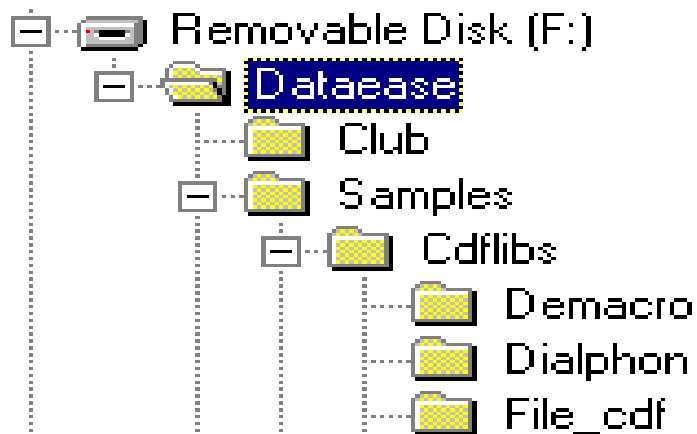


Figure 1: Folders and subfolders created by a default DfW install

Unfortunately, that means that the field in the CDF registration form in the DataEase application may run out of room for the whole name of the library (if you include the drive, folder and library name). The field in the Custom Functions form that records this information can contain a maximum of 40 characters. (**c:\dfw\samples\cdflibs\WizPrint\WizPrint.dll**, for example, is 44 characters long and won't fit into the field!).

As a result, I don't recommend that you use register the copies of the libraries in these subfolders.

Starting with DataEase 6.x, the standard Sapphire install procedure also placed all of the CDF libraries that shipped with the program in the DataEase program folder selected by you during the installation.

A Windows program will look for DLLs referred to by a program in the following folders (in order from first to last):

- The program folder
- The current folder

- The Windows SYSTEM folder
- The WINDOWS folder
- A folder specified in win.ini or the registry
- Folders in the PATH statement

(There seems to be some dispute as to order of the first two—some Microsoft documentation says “Current folder first, then program folder” whereas Sapphire claims it’s the other way around). In other words, if the CDF Registration form contains only the name of the library (and no drive and folder data) DataEase will look first in the DataEase program folder, then in the application folder. (I didn’t know this until version 6 was in beta, but it’s apparently been the case since DataEase for Windows first shipped. Amazing what’s not in the documentation, eh?)

Thus, in order to make the registration process simple, you could just ensure that all the CDFs are copied into each copy of DataEase on every hard drive---which is just fine if you are on-site, and if you need to change or update a .dll, you can run around to all the PCs on the network. Personally, I can think of more enjoyable ways to get my exercise. And you are setting yourself up for yet another source of “DLL Hell”.

(Or, I guess you could use some kind of scripting language to “push” the file changes out to the workstations, but many network administrators frown on scripting languages because of virii...).

Note as well, that if you will be using any of the Windows API calls directly (by registering functions from kernel32.dll, shell32.dll or whatever) that the search path rules above apply—you can simply enter the system dll by name.

If you start DataEase by clicking on the Start button on your PC and then open an application by navigating to it via the Open Application dialogue box, the DataEase program directory is both the program folder and the current folder as far as Windows is concerned. CDFs located elsewhere in your system must be explicitly pathed (e.g., drive and folder name must be spelt out in full) in the Custom Functions form.

b) Keep all the CDFs in their own folder under the DataEase program folder.

Another option is to flatten out the directory structure and create an extra directory (C:\DFW\CDF) and copy all CDF .dll files there. For example:

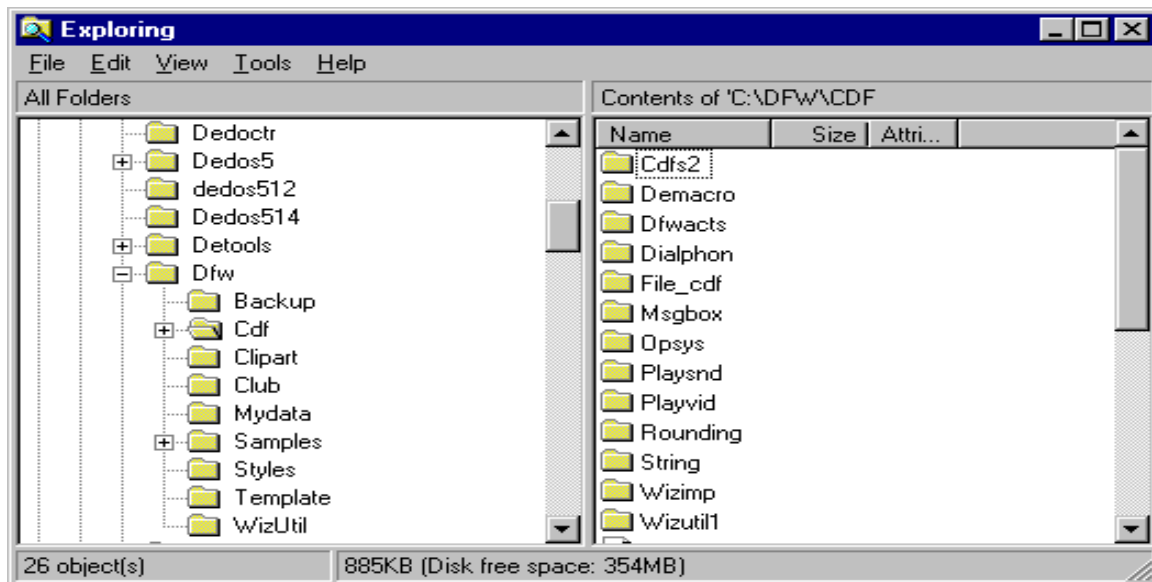


Figure 2: The “flattened” directory structure, e.g., “C:\DFW\CDF\...”

Not shown is that all the CDF .dll files are copied into C:\DFW\CDF.

After that, make another subfolder in the CDF directory (e.g., C:\DFW\CDF\LIBRARYNAME). Copy a backup copy of **LIBRARYNAME.dll**, plus any other files distributed with it into that folder—this is your backup copy in case anything happens to the working copies that the application(s) use.

This method was my preferred method up to and including DataEase 5.5. I’d put the CDF library directory on the server and ensure that all applications used these .dll’s. It certainly helped prevent “DLL Hell”.

However, with the arrival of DataEase 6, Sapphire changed its network install strategy. DataEase must now be installed on the local hard drive of each PC accessing shared data on the server, so there is no central repository any more. While you could still create a CDF directory on the server, there may be other reasons to use the other options.

If the CDF libraries on the server, you still must specify the drive, folder and library name when registering the CDF with your application and you are still subject to the 40-character limitation.

c) Copy all of the CDFs into the application (data) folder.

With this option, you simply place the CDF library CDFs into the folder with your data.

This one has the advantages of:

- Reduced running around.
- CDFs all in one place with the application when it is backed up or moved.
- Simplified registration within DataEase.

versus

- Multiple copies of the .dll's on the server if you have multiple applications.

However, if you do copy your CDF libraries to the data folder AND you don't explicitly identify the drive and folder in the Custom Functions Form, then you MUST start DataEase with a shortcut that points to the application instead of starting it from the Start button and navigating to the application via the Open Application dialogue box.

The command line of the shortcut must be something like:

```
D:\DFW6\DataEase.exe f:\modimp "Modify Import 32" "user" "user"
```

which registers **D:DFW6** as the “application” folder and **f:\modimp** as the current folder for the DLL search path.

d) Combine strategies (c) and (c).

In this approach, I keep the Sapphire-supplied libraries in the DataEase program folder (where they are installed by default by the installation program) and place any other ones (like the ones from ComputerWizard Consulting) in the application (data) folder.

When I was installing an application at a client's site with a network, I realized that this was a good solution as well, since they were going to install DataEase by physically running around to each PC and running the install there, rather than “ghosting” or “imaging” the new PCs from a reference machine. As result, I'd have to leave detailed instructions for the IS staff as to which files to delete from the DataEase program directory on drive C—and this got to be a bit too onerous.

This still leaves you with the possibility of “DLL Hell”, if Sapphire ships some updated versions of the libraries, but such is life!

As with option (c) you MUST start your databases with an explicit shortcut that points to the data folder, e.g. the command line of the shortcut must be something like:

```
D:\DFW6\DataEase.exe f:\modimp "Modify Import 32" "user" "user"
```

which registers **D:\DFW6** as the “application” folder and **f:\modimp** as the current folder for the DLL search path.

In summary:

Option	Pro	Con
a) DataEase Program Folder (default)	<ul style="list-style-type: none"> • Less work during installation, especially on a network. • Patches and updates from Sapphire that are looking for the CDF libraries will run without those annoying error messages about files not found. • CDFs don't have to be explicitly pathed (drive and folder spelt out), just named. 	Possibility of “DLL Hell” since DLLs are on each user’s PC.
b) Subfolder under DataEase program folder or on server.	Server option places DLLs in one location for easy administration.	<ul style="list-style-type: none"> • CDFs have to be explicitly pathed (drive and folder spelt out), just named. • Can reduce the path so that you keep within the 40-character limit.
c) In the database (application/data) folder.	<ul style="list-style-type: none"> • CDFs all in one place with the application when it is backed up or moved. • Simplified registration within DataEase (explicit pathing not required). 	<ul style="list-style-type: none"> • Multiple copies of the .dll’s on the server if you have multiple applications; upgrades and patching can be a problem. • If you don’t explicitly path your functions, then you MUST start the database from a shortcut that identifies the application instead of “generic” DataEase shortcut.
d) Combination of (a) and (c).	See pros of both options	See cons of both options

My preferred approach is (d).

2) Copy the DLLs to the appropriate folders depending on your strategy.

There is no “install” procedure for CDF libraries as far as Windows is concerned—you need only copy the **.dll** to a folder and you’re done.

3) Register the functions with DataEase.

Before continuing, make a backup of your application using a tape backup, a second copy elsewhere on your drive or network, a ZIP drive or use PKZIP.

Don’t use the DataEase backup utility unless you *like* taking risks with your life and data.

Make sure that nobody else is using the application until you are finished installing the library! (Locking the database can certainly help).

Once again from the help file:

Before you can use a CDF, you must register it with your DataEase Application—registration “plugs in” the function.

Once registered, Custom Functions work in exactly the same way as the built-in DataEase functions. So they can be used in Field Validations and Derivations, Button and Picture Actions, Filters, Procedures and Custom Menu Bars. They can not be used on a Menu Document.

From The DataEase CDF Help File
© Sapphire International

To use any CDF library with DataEase for Windows, you must add the function(s) contained in the **.dll** file to the application’s Custom Functions table. You must go through this exercise each time you install a library in **each** application you create. You can do this the hard way or the easy way. You can register some or all of the functions; I generally recommend installing all of them from each library once you decide on installing a library.

The Hard Way—One function at time and you do all the typing.

- 1) Start DataEase.
- 2) From the DataEase Main Menu, select **A**pplication, then **C**ustom **F**unctions.
- 3) Fill in the Custom Functions form, once for each function. See any documentation or help files for instructions.
- 4) When done, exit and restart DataEase.

Yeah right. If you wanted to work this hard for a living, you’d be programming in Access, right?

The Easy Way—Let DataEase do most of the work! (The non-Sapphire method)

You can import the data into the Custom Functions table quite easily. In fact, if you have one application already set up and working fine, it's a matter of 15 minutes (maximum!) to export the CDF records from one application and import them into another. As a developer, I keep a standard text file around of all the CDF libraries that I routinely install and every time I start a new project, I simply import this standard data file.

A file with all of the CDFs that ship with DataEase 6.x is available from Kingston & Co's website at:

http://www.kingstonco.com/dataease_6.htm

All libraries from ComputerWizard Consulting include a text file, **wizCDF.txt** that contain the functions for the library and can be used to import them into the Custom Functions form.

To import the functions into DataEase:

- 1) Start DataEase for Windows.
- 2) If you have not changed your Application Preferences to "Show System Tables" do so. Select **Application | Preferences** from the pull-down menu, and click on the **Show System Tables**: check box (the default on installation is to hide these tables):

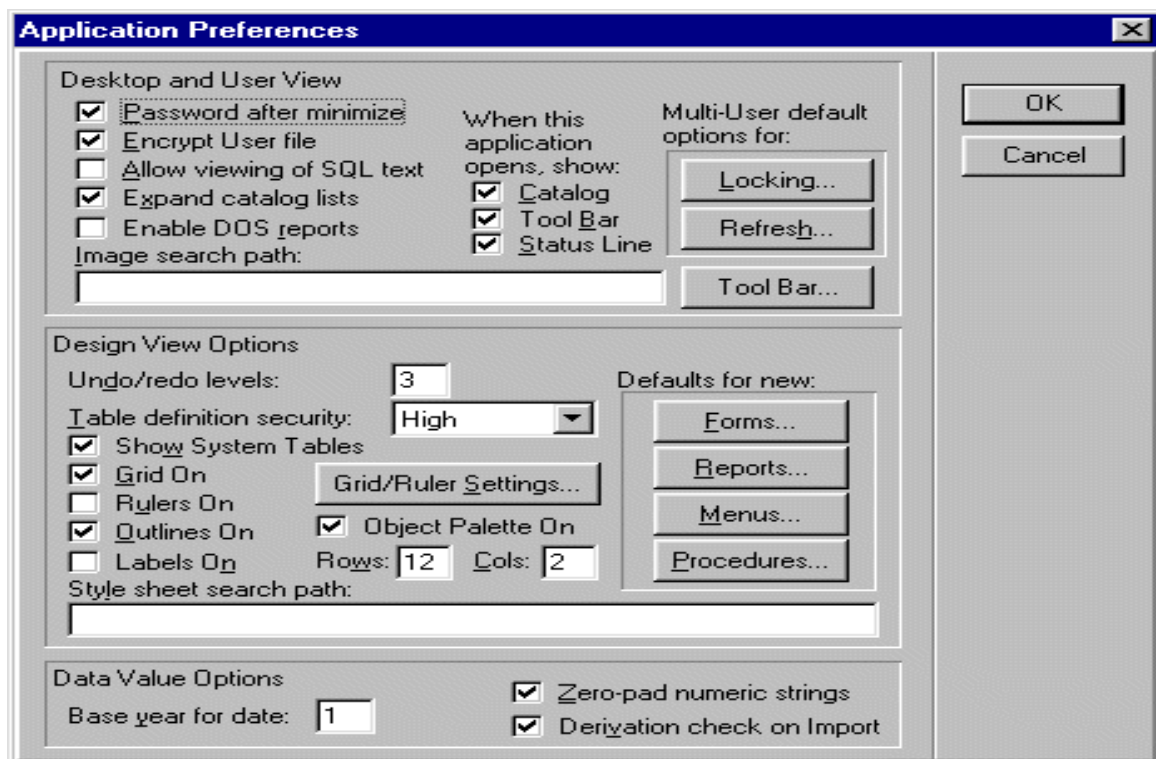


Figure 2: The DfW Application Preferences Dialogue Box, with the **Show System Tables** option checked; the default on installation is to NOT show the system tables.

Click OK when you're done.

- 3) From the DataEase Main Menu, select **File | Import**. In the Import Dialogue box, fill in the following information:

Field	Data
Destination File Name	Custom Functions
Import Format Type	Variable Length Text (Windows ANSI)
Import Source File Name	Click on the "Browse" button and locate the folder with the data file in it, then select that file.
File Organized by:	Field Name
Field Separator:	~ (default choice)
Record (Separator):	New Line (default choice)

Then click on the Run button.

- 4) Exit DataEase, then restart it. (Otherwise, you will not have access to the Custom Functions table for the next step.)
- 5) Restart DataEase. Reorganize the Custom Functions. Exit and restart it again.

At this point, you have entered the functions in this library into the Custom Functions table, If you have selected option (b) for the location of the libraries, you'll need to change the drive and folder names in the Custom Functions form. I've provided a DQL at the end of this article to make your life easier. If you do this, exit DataEase, restart it and reorganize the Custom Functions form, exit and restart again.

The Easy Way—Let DataEase do most of the work! (The Sapphire method)

Sapphire suggests another route—importing the functions one at a time from the CDF sample application, assuming that you've installed it.

Again, from the help file:

Filling in the form

As with all the CDF's included with DataEase, the quickest way to register a CDF in your own application is to cut and paste the appropriate CDF function from the CDF

Example Application. The process is as follows:

Open the sample CDF application—called CDF Library—which comes with DataEase. (The default path is DE6\Samples\CDFLibs\CDF Library). Open the Main Menu, press the button on the bottom of the screen labelled “CDFs SYSTEM FORM”. Press F3 until you find the function you want to use in your own application. From the menu bar select Edit>> Copy Record. Exit the Sample CDF application and access your own application. Open the CDFs System Form, Paste the record, and save it.

Note: You may need to change the contents of the CDF LIBRARY NAME field to make sure DataEase can find the DLL file.

From The DataEase CDF Help File
© Sapphire International

Personally, I think this is ***dumbest*** method of doing things ever suggested by Sapphire; it's slow and ponderous, and it assumes that you will only load into your application the functions required, and that you'll keep on going back to the well each time you need a function. It also assumes that you've installed the CDF Library sample application at a client's site (if you're a developer with multiple clients).

If you don't want to download the file from Kingston & Co, you can create it by exporting the data from the sample CDF application and use that file as the basis for your import.

A DQL to Change CDF Drives and Folders

If you have created an application that uses explicit drive and folder assignments for the CDFs and need to change them to something else when the application is deployed, you can use the following DQL. Just copy from here and paste it into a DQL, save it, check it (compile it) and run it.

This one contains most of the standard CDF libraries that ship with DataEase; you may need to edit it for your own install in case you have additional ComputerWizard libraries (or other libraries from other sources).

```
-----
-- Procedure:      a_Update_CDF_Paths
-- Purpose:        Edits the Custom Functions table to change any and all CDF
                  path
                  names
-- D/E Form?      No.
-- Called by:
-- Designed:      June2002, L. Fox, ComputerWizard Consulting
-- Revised:
-----

define temp "tCDFPath" Text 20 .

tCDFPath := "" . -- don't put the library name here; see below!
            -- leave blank for no drive and folder, otherwise enter one.

-- standard CDF libraries that ship with DataEase

modify records in System Custom Functions with Library Name = "*cdfs2.dll"
    Library Name := jointext(tCDFPath, "CDFs2.dll") .

modify records in System Custom Functions with Library Name = "*demacro.dll"
    Library Name := jointext(tCDFPath, "DEMacro.dll") .

modify records in System Custom Functions with Library Name = "*dfwacts.dll"
    Library Name := jointext(tCDFPath, "DFWActs.dll") .

modify records in System Custom Functions with Library Name = "*fileexec.dll"
    Library Name := jointext(tCDFPath, "FileExec.dll") .

modify records in System Custom Functions with Library Name = "*memarr32.dll"
    Library Name := jointext(tCDFPath, "MemArr32.dll") .

modify records in System Custom Functions with Library Name = "*msgbox.dll"
    Library Name := jointext(tCDFPath, "MsgBox.dll") .

modify records in System Custom Functions with Library Name = "*osfunc.dll"
    Library Name := jointext(tCDFPath, "OSFunc.dll") .

modify records in System Custom Functions with Library Name = "*playsnd.dll"
    Library Name := jointext(tCDFPath, "PlaySnd.dll") .

modify records in System Custom Functions with Library Name = "*playvid.dll"
```

```
Library Name := jointext(tCDFPath, "PlayVid.dll") .  
modify records in System Custom Functions with Library Name = "*rndwin.dll"  
Library Name := jointext(tCDFPath, "RndWind.dll") .  
modify records in System Custom Functions with Library Name = "*strfunc.dll"  
Library Name := jointext(tCDFPath, "StrFunc.dll") .
```