

Moving Large Multi-DLL Applications to Clarion 7

By: Abe Jimenez

Introduction:

Like most of you, I've waited for years to get my hands on Clarion 7. Now that I have it, I needed to convert my Clarion 6.3 applications and start using my new tool. Most of the applications I work on are multi-dll solutions. Soft Velocity provides a multi-dll example application, and in one of the videos (I think it was Clarion Live) they showed how to go about converting it to Clarion 7. I tried to convert my applications using the same technique they used, but was un-successful.

The problem was that my applications were much bigger than Soft Velocity's example. My code is much more complex, and I use lots of 3rd party products. Recently I took a different approach at converting the same set of applications, and this time I succeeded. In this document I will show you how I did it.

First Things First

It's important to prepare for a project of this type. If you are not careful you can end up with applications that can't be opened in Clarion 6 anymore and that won't compile in Clarion 7. This is because once you open either an application or a dictionary in Clarion 7 it goes through a conversion to a newer format. Clarion does save the old version of the files with a new extension (.6APP or .6DCT), but I consider it too risky to do any kind of a conversion without taking steps to make sure I can easily cancel and go back to what I had. So my first recommendation is that you copy your application folder and use the copy to do the conversion. If everything fails, you can still use C6.X or whatever version you are using until the issues are resolved.

One of the options you have to consider is to convert your application and dictionary files, but to continue to compile in Clarion 6. At first this looked like a good intermediate step. I would be able to continue to use my existing third party templates and the conversion would have been much easier. Unfortunately, this did not work for me. It turns out that the new IDE enforces the template coding rules a lot more strictly than the old one and some of the templates I used in C6 cannot be registered by the C7 IDE. If I have to get new templates, it makes better sense to me to get the Clarion 7 version.

Unfortunately some of my 3rd party vendors have not made a Clarion 7 version of their products available. I waited what I consider a long time before giving up on them. Clarion 7 went gold over a

year ago and Clarion 7.1 is the second gold version of C7.X. If a vendor still does not have a Clarion 7 version of their products, I have to assume they are not making one.

Preparing Clarion 7

I obtained all C7 versions of the templates I use and registered them in the IDE. Then I converted and compiled the Clarion 6 example applications for each of the 3rd party products. This assured me that if I had conversion problems with my programs, they were because of my code and not because of the new version of the 3rd party product.

Preparing the Clarion 6 Applications for Conversion

Since I couldn't get C7 versions of all the 3rd party products I use in C6, I had to find a way to replace their functionality. I did this in C6 to make sure that I was starting my C7 conversion with applications that should convert properly. When I got done with this step, I had C7 versions of all the templates I was using in C6.3.

The Conversion Process

Now I started to do the C7 conversions. In my previous attempt, I found that for large solutions it's not a good idea to try to convert everything at once. If you have even a few errors that will cause some of your DLL's not to compile, the other applications that reference the DLL will fail compilation as well. It makes more sense to me to convert the applications one at a time so that the problems with one application would not affect the next one.

I started by opening my application in Clarion 7. There is an option in the file menu to open a project or application (see figure 1). I started with the root or global data application and then proceeded to do each of the other applications. The order in which you do the conversions doesn't really matter at this time.

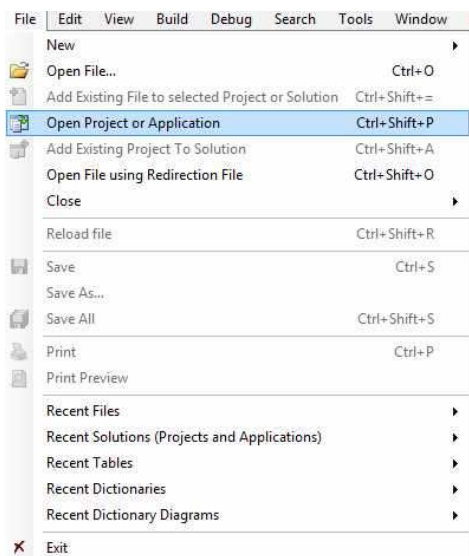


Figure 1

This opens up the Windows file dialog and lets you select the application you want to convert. When you select a file, you will see the message displayed in figure 2.

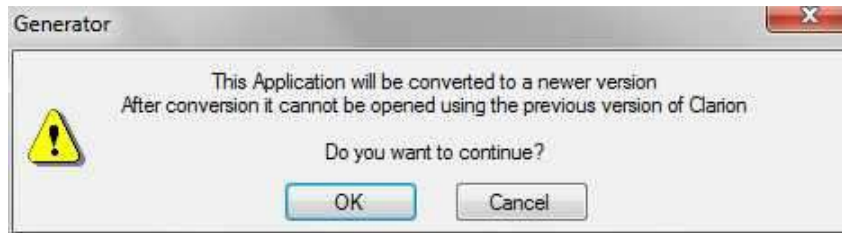


Figure 2

If the dictionary has not been converted yet, the next message was displayed as shown in figure 3.

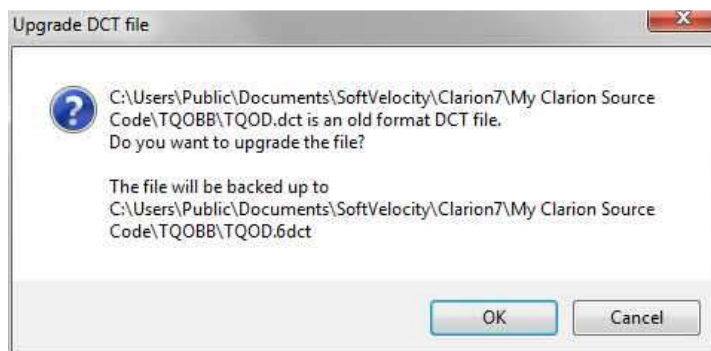


Figure 3

Click OK here too. This is when the conversion really starts. Clarion 7 creates copies the C6.X (or previous) app and dictionary to files with the same name but with extensions of .6APP and .6DCT, converts the application and dictionary to the Clarion 7 format, and creates .sln and .cwproj files.

If your application calls procedures from another dll you will see the message shown in figure 4. This will probably not happen with the root application, but as you repeat these steps for other applications it is important that you click NO for now. Clicking YES will add all the referenced applications to your solution. As I said before, you want to resolve your problems one application at a time.

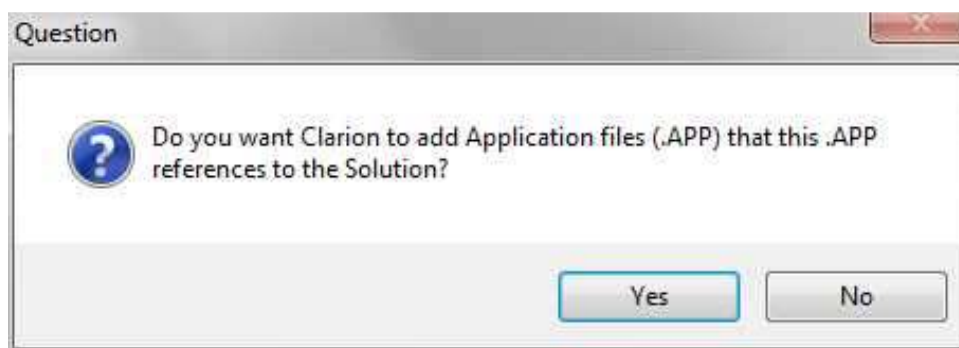


Figure 4

Once the conversion completes, Clarion will display any conversion error messages. There should not be many errors since you have taken steps to minimize the problems. But if you have conversion problems here, you have a choice of fixing them in Clarion 6 and reconverting, or just fixing them in Clarion 7.

If you choose to go back to C6 to fix your conversion errors, you need to do the following.

- Delete the app, sln, and cwproj files.
- If the dictionary is being converted, you can delete the dct file as well
- Rename the .6app and .6dct back to the original names.
- Use C6 to fix your problems
- Start the conversion process for the app again.

Compiling and Making your Application

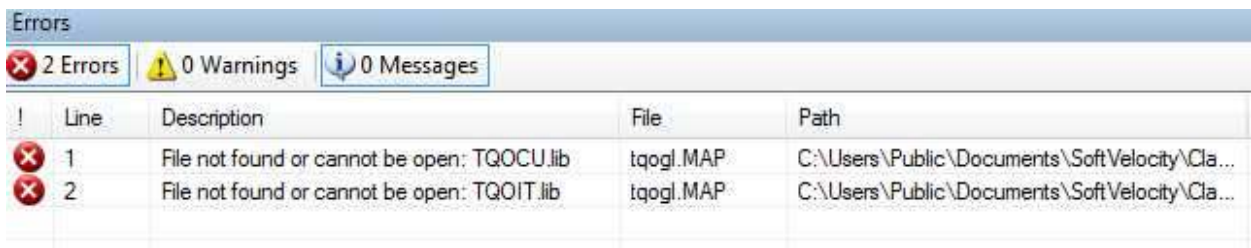
When I started this process I was using C7.0.5768 which did not provide application recovery after a crash of the IDE and the IDE crashed a lot. After 7.1 SV provided application recovery, but many people have turned it off to make Clarion run faster. I'm not having speed issues with C7.1, so I left application recovery on. If you are not using application recovery, it's important to save your work here.

Figure 5 below shows the icon in the IDE where you can generate and compile the current application.



Figure 5

At this point, you should not expect any application that references an external DLL to compile. This is because you have not included the external DLL's in your solution. Figure 6 shows the messages you will get when you compile an application that references another application that has not yet been compiled.

A screenshot of the 'Errors' window in the IDE. The window title is 'Errors'. It shows a summary of 2 Errors, 0 Warnings, and 0 Messages. Below the summary is a table with columns for Line, Description, File, and Path. Two errors are listed: Line 1: File not found or cannot be open: TQOCU.lib, File: tqogl.MAP, Path: C:\Users\Public\Documents\SoftVelocity\Cla...; Line 2: File not found or cannot be open: TQOIT.lib, File: tqogl.MAP, Path: C:\Users\Public\Documents\SoftVelocity\Cla...

!	Line	Description	File	Path
✖	1	File not found or cannot be open: TQOCU.lib	tqogl.MAP	C:\Users\Public\Documents\SoftVelocity\Cla...
✖	2	File not found or cannot be open: TQOIT.lib	tqogl.MAP	C:\Users\Public\Documents\SoftVelocity\Cla...

Figure 6

If you get compile errors other than the lib file not found errors, you can fix them in either Clarion 6 or Clarion 7. To go back to Clarion 6 you can follow the same steps listed earlier in this article. At this point, it will probably be easier to fix most problems in Clarion 7, but I did run into some issues where I decided it was easier to just go back and fix them in C6. For example, I kept finding projects attached to my application that I could not delete. In those cases, I went back to Clarion 6, deleted the project and

saved the application without compiling. Then I started the conversion for that application again and the error went away.

Building the Multi-App Solution

Now that you have made sure the individual applications have been converted and have no compile errors, it's time to bring it all together into one solution. This is the solution you will use in the future to work on this set of applications.

You probably have a main exe that calls the procedures in most of your DLL's. For that main application you want to delete the sln and cwprj files. Be sure you do not delete the app file. Then open the application in the IDE. You will see the window in figure 4 above. Answer YES this time.

This will create new project and solution files. This time the solution will contain the exe's app and all the referenced DLL's. You will be able to see the components for the solution in the solution explorer window in the IDE.

For some solutions you will need to add other applications that are not directly referenced by your main executable. If you right click on the Solution Items tree in the solution explorer you will see the menu in figure 7. This is where you can manually add app files to the solution.

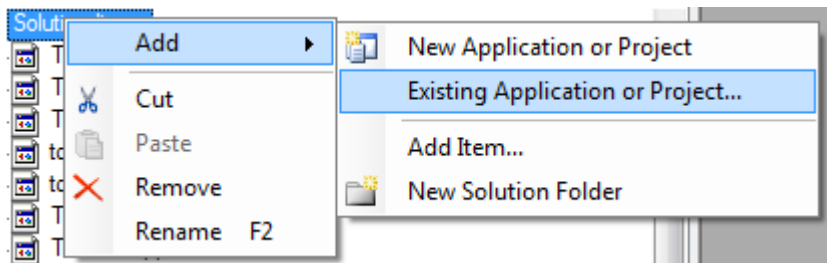


Figure 7

Now you are ready to compile the complete solution. Figure 8 shows where to go in the applications pad. This will regenerate and compile every application in the solution. In theory, you should not get any errors, since you have already taken care of all the problems in each of the individual applications.



Figure 8

Circular References

A circular reference is when application A references application B and application B references application A. It can even get worse than that. Application C can reference both A and B and B can reference C and so on. Clarion has allowed this for a long time, even though it has never been a

recommended practice. In earlier versions of C7 this was not allowed, and in fact there are a number of articles in Clarion Magazine explaining how to identify and remove these references.

Fortunately, Soft Velocity re-enabled this feature. Without it, it would have been necessary to re-distribute the procedures in your applications in C6 and get rid of all circular references before starting the conversion process. Although applications with circular references will compile in the newer versions of Clarion 7.X, it may be necessary to compile your solution multiple times to get all the applications to compile successfully. This was the same in Clarion 6.X.

On the first compile, 8 of the applications in my solution had errors as shown in figure 9 as a result of the circular references.

```
Applications processed without errors: (17)
tqoms
TQODI
tqoib
tqosw
tqodu
tqosu
tqopp
tqobk
TQOI2
TQOSC
TQOAP
TQOBA
TQOGV
TQORP
TQOPR
TQOAR
TQOGL

Applications with errors: (8)
TQOVN
TQOCU
TQOIT
TQOFL
TQOFN
TQOCO
TQOSE
tqoma
```

Figure 9

After compiling a second time without making any changes, the errors went away as shown in figure 10.

```
-----  
Batch Processing Summary  
Generation and BatchCompile was finished at 11/4/2009 12:12:34 AM and took 00:18:01.5998640  
  
Applications processed without errors: (26)  
tqocd  
tqoms  
TQODI  
tqoib  
tqosw  
tqodu  
tqosu  
tqopp  
tqobk  
TQOI2  
TQOSC  
TQOAP  
TQOBA  
TQOGV  
TQORP  
TQOPR  
TQOAR  
TQOGL  
TQOVN  
TQOCU  
TQOIT  
TQOFL  
TQOFN  
TQOCO  
TQOSE  
tqoma  
  
Applications with errors: (0)  
--None--  
|
```

Figure 10

Running and Testing your Program

You are probably tired now and you think you are done because everything compiled. But you really need to test your program. Since you haven't changed a lot of code, your functionality should not have changed all that much, but there are a few things to look out for.

Clarion 7 windows and reports don't look exactly like they did in Clarion 6 even if the code is the same. Some controls will be wider for example. So you need to open each window and run each report to make sure it still looks fine. If you had controls that were close to each other, they might now overlap.

The templates you are using in C7 may not be the same as the ones you were using in C6. So the generated code might be different. Remember that some vendors had trouble getting their Clarion 6 templates in the new IDE. You might have installed the newer version of the third party products in the new IDE but left your Clarion 6 environment unchanged.

If you made changes to your code as a result of a compile or conversion error, you need to test those too.

You Are Done

By the time I got to this point I was exhausted, but very happy with my results. I did the preparation work over a long period of time so I'm not sure how long it took. Luckily, this is a one-time thing.

The actual time it took me to convert my 20+ app solution was about 6 hours. This includes solving the errors I got during import and compiling. Testing the new version of my application took a long time, since I had so much to test. If your application is in use by a large user base, you might want to make your new version available to a small group of users for testing.

I have now followed these steps for about 40 solutions (over 700 apps). They are all tested and deployed to my customers. Of course your results may vary since we all do things differently. So please use this document only as a guideline.