Clarion Magazine

Clarion News

- » PrintWindow Build 108
- » PowerOffice Christmas Holiday
- » PowerXP-Theme Price Increase Jan 1
- » xButtonTimeSelect Released
- » Snazzy Listbox v1.24
- » SetupBuilder 5.2 Build #1354
- » ProcedureNotes Template Sale
- » BST Ver 4.21 Patch
- » EasyOpenOffice 1.01
- » Fomin Report Builder 3.00, Fomin ODBC Report Storage
- » Lesley Dean On Leave
- » SimPageofPage Template Updated
- » Year-end 2005 UltraTree savings
- » UltraTree 8.5
- » FullRecord v1.16
- » cpTracker Professional 1/2 Price Sale Until Dec 31
- » PuterBuddy December Pricing
- » Academic Version of Clarion Available
- » C7/Clarion.NET Alphas In January, Clarion.NET To Have WebForms and WinForms
- » FullRecord 1.15
- » Clarion Web Hosting Running IPD 2.2
- » Whitemarsh Data Interoperability Workshop
- » DCT2SQL Templates Updated
- » Template Finds Orphaned Embeds
- » CDC Week #13 Winner

- » Seven New Ready-to-go Logo Packages
- » Fenix Becomes GenWise
- » Fenix 2.00 Nearing Gold Release
- » PowerOffice And 9049
- » Changes At ClarionTools.Com
- » Capesoft Accessories All 9049 Friendly
- » New Free Clarion Wallpaper
- » PowerXP-Theme 2.2
- » RPM, AFE, PNet For 9049
- » GWBHTMLHelp - HTML Help From Your Clarion app
- » Gitano Utilties For 9049
- » PrintWindow Build 105
- » Gitano Winter Sale
- » CDC Week #12 Winner

[More news]

Podcast

[Track lists, more podcasts]

Latest Free Content

- » Holiday hours and Christmas greetings!

[More free articles]

Clarion Sites

- » BoxSoft (a.k.a. Mike Hanson)

- » The Clarion Connection

Clarion Blogs

- » John Griffiths

Latest Subscriber Content

PDF for November 2005

All Clarion Magazine articles for November 2005 in PDF format.
Posted Tuesday, December 06, 2005

Modifying The Frame Background - A 10 Minute Template

Someone asked recently on the newsgroups if there was an easy way to change the background color of an application frame. This is a question Geoff Bomford frequently asked himself, and never found a satisfactory answer. He happened to find a solution that was good enough to turn into a simple template. Part 1 of 2.
Posted Thursday, December 08, 2005

Hello, Server?

If Henry Plotkin's application runs across a network, he wants to know if that network is working or not. If the network drops while the program is trying to access files, the user might get an error message. Or they might not. Or they might get a "Save As" dialog, as happened recently. If they do get a message, it is unlikely that they will comprehend it. Something more user friendly is in order.
Posted Monday, December 12, 2005

Using GetLastError and FormatMessage to Check the Network

As Henry Plotkin discovered in his previous article, IsDestinationReachable is a less-than-perfect API call for detecting network problems. In this installment, Henry expands his analysis with the GetLastError and FormatMessage calls.
Posted Thursday, December 15, 2005

Modifying The Frame Background - A 10 Minute Template, Part 2

Someone asked recently on the newsgroups if there was an easy way to change the background color of an application frame. This is a question Geoff Bomford frequently

asked himself, and never found a satisfactory answer. He happened to find a solution that was good enough to turn into a simple template. Part 2 of 2.
Posted Friday, December 16, 2005

IsNetworkAlive: When You Don't Need to Know *That* Much

Steve Parker's colleague Henry Plotkin recently examined ways of checking whether a client PC was able to talk to another, named, PC on the network. Henry liked the IsDestinationReachable API, but as Steve shows, IsNetworkAlive still has its uses.
Posted Friday, December 23, 2005

Translating Mapped Drives To UNC

Steve Parker recently objected to having to translate mapped drive letters to UNC names in order to check whether or not network resources were available. Paul Attryde, with his characteristic unstinting help, pointed Steve to a fairly easy and very solid solution.
Posted Friday, December 23, 2005

Holiday hours and Christmas greetings! (free article)

A note on our holiday hours, such as they are, and our best wishes to everyone for the Christmas season, or for whatever festival of peace and light you may enjoy at this time of year!
Posted Saturday, December 24, 2005

Clarion Apps Can Be Sexy!

Most of us think of Clarion applications as showing a bit on the dowdy side. But as Colin Wynn demonstrates, with relatively little work you can give your applications a consistent and attractive user interface.
Posted Friday, December 30, 2005

[Last 10 articles] [Last 25 articles] [All content]

Printed Books & E-Books

E-Books

E-books are another great way to get the information you want from Clarion Magazine. Your time is valuable; with our e-books, you spend less time hunting down the information you need. We're constantly collecting the best Clarion Magazine articles by top developers into themed PDFs, so you'll always have a ready reference for your favorite Clarion development topics.

Printed Books

As handy as the Clarion Magazine web site is, sometimes you just want to read articles in print. We've collected some of the best ClarionMag articles into the following print books:

- Clarion 6 Tips & Techniques Volume 1 - ISBN: 0-9689553-8-X
- Clarion 5.x Tips and Techniques, Volume 1 - ISBN: 0-9689553-5-5
- Clarion 5.x Tips and Techniques, Volume 2 - ISBN: 0-9689553-6-3
- Clarion Databases & SQL - ISBN: 0-9689553-3-9

We also publish Russ Eggen's widely-acclaimed Programming Objects in Clarion, an introduction to OOP and ABC.

From The Publisher

About Clarion Magazine

Clarion Magazine is your premier source for news about, and in-depth articles on Clarion software development. We publish articles by many of the leading developers in the Clarion community, covering subjects from everyday programming tasks to specialized techniques you won't learn anywhere else. Whether you're just getting started with Clarion, or are a seasoned veteran, Clarion Magazine has the information *you* need.

Subscriptions

While we do publish some free content, most Clarion Magazine articles are for subscribers only. Your subscription not only gets you premium content in the form of new articles, it also includes all the back issues. Our search engine lets you do simple or complex searches on both articles and news items. Subscribers can also post questions and comments directly to articles.

Satisfaction Guaranteed

For just pennies per day you can have this wealth of Clarion development information at your fingertips. Your Clarion magazine subscription will more than pay for itself - you have my personal guarantee.

Dave Harms

Clarion Magazine

Clarion News

Search the news archive

Data Equity Assistants Release 2.00.00.26

There are new updates available for Dictionary Assistant and Application Assistant. The final Dictionary Assistant release is expected during the coming week so you can consider this a release candidate. If you wish to update your installation to this release use the "Check for Update" option on the help menu or as always you may download the product from the web site.
Posted Monday, January 09, 2006

PrintWindow Build 109

New in PrintWindow Build 109: You can choose to print the selection bar on listboxes, marking the currently selected record. This is specially important when you have child boxes, as the records shown will be those related to the parent listbox, therefore the need to know which record is selected in this one. Compatible with Clarion 6.3 (9050).
Posted Monday, January 09, 2006

Enabling Simplicity Sale Extended To January 12

Enabling Simplicity's 2005/2006 holiday season sale has been extended five days, and will end on January 12, 2006.
Posted Monday, January 09, 2006

EasyCOM2INC_WMI 1.01

New demonstrations in EasyCOM2INC_WMI 1.01: How to read system registry keys on

local and remote computer; How to obtain a list of programs installed using Windows Installer, and how to Uninstall them; How to obtain required information using WQL (a subset of SQL); How to use asynchronous techniques and to be notified of their progress. Windows Management Instrumentation (WMI) is a component of the Microsoft® Windows® operating system that provides management information and control in an enterprise environment. Using industry-wide standards, WMI allows managers to query and set information on desktop systems, applications, networks and other enterprise components. Developers can use WMI to create event-monitoring applications that alert users when important incidents occur. WMI also offers a variety of programming interfaces, such as C++, ODBC, Microsoft® Visual Basic®, or HTML, that developers can use to further tailor their management applications. Finally, WMI integrates with other Windows components, such as the Active Directory, to allow for a unified management experience. System requirements: Clarion 6.1 9033 or higher; EasyCOM2INC 2.05 or higher installed; Windows Management Instrumentation (WMI) installed. Price: $30.
Posted Monday, January 09, 2006

ComSoft Updates

BRT 1.7 has been released, with new support for frameless apps. BST 4.22 added an embed to automate Holiday Class in Wall Calendar and Weekly Planner. BFET 2.1 has a new embed button. BsIdent 1.1 is a freeware program to remove Idents from an exported TXD File.
Posted Monday, January 09, 2006

Aussie DevCon Registrations Now Open

If you have ever wanted to visit the Great Australian Outback, or the rugged beauty of the Kimberley, taste the best wines in the world, or even just surf and lie on the great beaches of Western Australia, or maybe visit the tourist meccas of Sydney and the Gold Coast, or take a trip back in time to the widerness regions of Tasmania, perhaps wrestle with crocodiles and mosquitoes in the Northern Terrirory - you can do all of that *after* you come to the Aussie DevCon in Melbourne. The conference will be held Friday 28th April to Sun 30th April 2006 in Geelong, one hour out of Melbourne, with direct connections at the airport. Book soon, places are filling fast.
Posted Monday, January 09, 2006

CDC Week #17 Winner and Grand Prize Winner

The First Place Prize for the final week of the Clarion Developers Challenge football pool

is Is (BoTran) Translation Manager $199, won by Ken Bame. The grand prize winner, with 141 correct selections over 17 weeks, is Dave Bratovich. Dave wins a five user license to Gary James' Organize 365.
Posted Monday, January 09, 2006

Epison Concepts Becomes Epsilon Network, LLC

Epsilon Concepts is now Epsilon Network, LLC. Epison's web sites include: Epsilon Consult, for Internet Consulting; Epsilon Hosting, for domain registration and hosting; Epsilon Concepts, for Design & Development; Epsilon Message, for copywriting and content services; Epsilon Promote, for Internet Marketing; Epsilon Software, for Software Products & Services; Epsilon Directory, a free online directory, and Epsilon Support, a support portal. To celebrate our re-launch and the onset of a New Year, Epsilon will give any Clarion Developer or Industry Professional 25% off hosting packages and a free domain name with any purchase. Epsilon will also give a 25% discount on any package offered on www.epsilonconcepts.com and www.epsilonpromote.com.
Posted Monday, January 09, 2006

PrintWindow Build 108

New in PrintWindow build 108: Now you can expand a listbox horizontally to show all its columns; For "manual" or "real size" scale, now if the printout don't fit in the report width, it will horizontally split it using as many pages as necessary.
Posted Wednesday, December 28, 2005

PowerOffice Christmas Holiday

The PowerOffice office will be closed until January 2nd, and replies to emails may be delayed.
Posted Wednesday, December 28, 2005

PowerXP-Theme Price Increase Jan 1

As of January 1, 2006 the price for PowerXP-Theme will increase from $99 to $249. Clarion now has XP theme support for buttons and text fields; PowerXP-Theme adds XP theme support to entry fields, droplists, dropcombos, listboxes and spinboxes, sheets and EIP controls.
Posted Wednesday, December 28, 2005

xButtonTimeSelect Released

New from SealSoft, xButtonTimeSelect is a control template (button) that lets you insert time values into any entry field. Support SingleExe, MultiDll (Local Mode, Standalone Mode), 32-bit. Compatible with Clarion 6.2 (build 9047-9049), Clarion 6.1 (build 9034), Clarion 5.5 and Clarion 5. Price is $29.
Posted Wednesday, December 28, 2005

Snazzy Listbox v1.24

New features in Snazzy Listbox 1.24: Group headers and multiline groups are now handled; Drop/Combo listboxes are now handled. Available at the Download Center.
Posted Wednesday, December 28, 2005

SetupBuilder 5.2 Build #1354

Click on "Check for updates" in the "Help" menu to auto-update your current SB5 version. You can also download the full SetupBuilder 5.2 Build #1354 install image.
Posted Wednesday, December 28, 2005

ProcedureNotes Template Sale

Castle Computer's ProcedureNotes template is now on sale until Dec 31, 2005, normally $29, now only $21. Template highlights: Keep detailed notes about your application; Keep detailed notes about each procedure; Keep a To Do List both globally and for each procedure; Keep a revision history both globally and for each procedure. There are also two utility templates: Print the notes and information about your app; Print just the notes, To Do and revision history.
Posted Wednesday, December 28, 2005

BST Ver 4.21 Patch

A BST Ver 4.21 patch is now available. Includes one bugfix and more embeds.
Posted Wednesday, December 28, 2005

EasyOpenOffice 1.01

EasyOpenOffice 1.01 is now available. New methods include: SetPageMargins (EasyWriter) - sets page margins; AddGraphic (EasyWriter) - adds a picture to the

document; SetPageProps (EasyWriter) - sets page layout of Writer document; SetPageProps (EasyCalc)- sets page layout of Calc spreadsheet. New templates: PageSetup code template (EasyWriter) - sets some of the page properties; - AddGraphics code template (EasyWriter) - adds a picture into the document. Free for all registered customers. EasyOpenOffice is a set of classes and templates allowing you to exchange data between Clarion applications and OpenOffice Calc and Writer. Price: $149
Posted Wednesday, December 28, 2005

Fomin Report Builder 3.00, Fomin ODBC Report Storage

Fomin Report Builder 3.00 is now available. This is a major update. As well, the new Fomin ODBC Report Storage uses direct ODBC API calls to store FRB report layouts on any ODBC connected SQL server.
Posted Wednesday, December 28, 2005

Lesley Dean On Leave

CapeSoft's Lesley Deanwill is on leave until about the first week of Jan. Please send all support requests to support at capesoft dot com.
Posted Wednesday, December 28, 2005

SimPageofPage Template Updated

This latest release of the SimPageofPage template now allows programmers to change the English words 'Page' and 'of' to the language of their choice.
Posted Wednesday, December 28, 2005

Year-end 2005 UltraTree savings

Get UltraTree Lifetime Premium Edition bundled with Up & Up for a saving of $200 off the combined regular prices. Get UltraTree Lifetime Premium Edition only and save $100 off the regular price.
Posted Wednesday, December 28, 2005

UltraTree 8.5

Version 8.5 of UltraTree has been released. Some new features in version 8.5 include computed fields, reset fields, the ability to conditionally suppress sections of the tree, and customized popup support for different sections of the tree. Many exisiting features have

been tweaked and enhanced.
Posted Wednesday, December 28, 2005

FullRecord v1.16

FullRecord lets you track all changed data, so you know exactly who changed what and when. Among other things you can use this data to recover lost information, analyze operation sequences to optimize the system work, find operational errors and educate the user to improve the use of the system.
Posted Wednesday, December 28, 2005

cpTracker Professional 1/2 Price Sale Until Dec 31

cpTracker Pro is contact and project management software that has many features of particular interest to software developers and software entrepreneurs. To get the discount, select the Custom Solution option. Enter $49.00 USD. Regularly $99. For every two user licenses purchased, get the third free.
Posted Wednesday, December 28, 2005

PuterBuddy December Pricing

PuterBuddy has recently been released by Comsoft7 and PuterWare. Regular price is $79, but for December you can get PuterBuddy for $49. The real assets in the program are the data tree and WebLog features. PuterBuddy is password protected, and data is encrypted in tps files. Included are a Personal Monthly Calendar that can show recurring appointments, holidays, and birthdays from the address book. The data tree is a free form tree that can be used for almost anything, especially code snippets. It has a drag and drop feature that allows re-organizing the tree/menu structure easily by moving all the children when needed to other levels in the tree. Check out the screen shots and the online help files. Uses Makeover so you can change the appearance if you wish. Full featured 30 day evaluation, record limited after that for unregistered users. Branding on screens and reports. Unlock Key sent to email address. For special developers pricing, use the Comsoft7 Prices/Download page, scroll down to the "Special Order Block $25.00". Select "Coupon Code" for PO Type, and Include the "Name to Register PuterBuddy to" in the "Order For" text box along with any other notes. Indicate here any special Email address if needed other than PayPal.
Posted Wednesday, December 28, 2005

Academic Version of Clarion Available

A no-charge version of Clarion is available for academic use, where Clarion is installed in the university lab. Individual student copies are available at low cost, and instructors receive CD-based training courses.
Posted Thursday, December 08, 2005

C7/Clarion.NET Alphas In January, Clarion.NET To Have WebForms and WinForms

Replying in a thread about the Planet Clarion podcasts, Bob Z stated yesterday that alpha testing for Clarion.NET *and* C7 will begin with a select group in January. Z also said that Clarion.NET "can be used" with WebForms *and* WinForms.
Posted Wednesday, December 07, 2005

FullRecord 1.15

New in FullRecord 1.15: Local alternative to global generation for "Inspect" control template (If global option is not active, then local generation will take place, instead of issuing an error); Fix in global "inspect" control template - support for inspecting other audit files than the global declared one (for example the historic audit file).
Posted Wednesday, December 07, 2005

Clarion Web Hosting Running IPD 2.2

Clarion Web Hosting has updated its servers to run version 2.2 of the IP Driver, and the IP Driver demo has been recompiled using 9049 and version 2.2 of the IP Driver. A test app is available to demonstrate the speed gains, and which also demonstrates a further enhancement to make multiple browses on a window run faster still. This enhancement is the menu item "Invoices... (Enhanced Page Loaded)". Richard Rose also reports testing with a database of four or five thousand news group postings from Comp.Lang.Clarion to further prove the viability of this technology. A 10mb TPS file with over 15,000 names and addresses can be sorted and located on in a matter of a few seconds over an ADSL 1Mb feed. You can also leave any feed back in the demo app if you want.
Posted Wednesday, December 07, 2005

Whitemarsh Data Interoperability Workshop

Whitemarsh Data Interoperability Workshops () address the problem created when organizations develop a number of different business information systems that have overlapping data. A common solution to this data redundancy and synchronization problem is to create extracts-transforms-and-loads of data from one database and put the data into another database. While still redundant, it is now synchronized. This is acceptable if the quantity of databases is small. The problem grows unacceptably when the quantity of systems gets beyond 5 or 6. All the combinations and permutations of the inter-data transfers quickly consumes too much time and resources. Here are two possible solutions: First, build a much bigger database that encompasses all the data. Second, build a shared-data database that all the systems access to store and/or retrieve the shared data. Under either scenario, the very first step is to "know your data." That's the value provided by the Whitemarsh Data Interoperability Workshops, which provide organizations a methodology and strategy to integrate data across legacy systems. These workshops provide hands-on activities wherein common shared data is discovered and database designs about the shared data are created. Workshop attendees return home not only ready to put to work what they learned in class, but with all the necessary tools to carry out that work. Students return with a production class CASE/Repository system that can import SQL schemas, model shared data, set the shared data within contexts of missions, organizations, and functions, and finally generate SQL schemas about the discovered shared data. For more information, contact Mike Gorman at 1-301-249-1142 and/or through email at mmgorman@wiscorp.com. Workshops are scheduled for February and March, but immediate needs can also be accommodated. Workshop attendance is strictly limited to 15 so that there is maximum time for individual attention.

Posted Wednesday, December 07, 2005

Clarion Magazine

Modifying The Frame Background - A 10 Minute Template

by Geoff Bomford

Published 2005-12-08

Someone asked recently on the newsgroups if there was an easy way to change the background color of an application frame. This is a question I frequently asked myself, and never found a satisfactory answer, so I was interested in the suggestions that were posted in response. I happened to find a solution that was good enough to turn into a simple template. In this article I'll discuss the solution and the steps I went through to create this "10 minute template".

Coloring the frame background

Technically, the only way to change the background color of a Windows Frame is to mess with Windows API functions that change the coloring within Windows. The simplest way to change the background color of a frame is to specify a background image and set it to "tile" in the window properties. In this way a small square image can be used and the background frame will indeed be set to whatever color the image has. Using long, thin tiled images can create interesting effects, as can images with textures. The drawback to this technique however is that the background frame is still relatively boring, with no option to place anything else on the frame, such as a logo or copyright statement. (See gwb_fb1.app in the downloadable source for an example.)

A similar technique is to use a larger image and either stretch or center it in the frame. Depending on the technique chosen you can end up with a centered image surrounded by the gray frame, or a larger image that might change shape if the end user adjusts the resolution of their monitor. It is still a fairly rigid solution, but it gives you the chance to have your logo on the frame background, or to allow your end user to customize the frame with their logo, by specifying the background image at runtime.

The third solution is to use a technique similar to a splash window, but never close the splash window. This has the benefit of giving you a free rein as to what you put in the window; logos, images, copyright statements; and, the background window can be any color you or your customer like! Maximize the window, remove the title bar and system menu, and it will actually look like the frame background. There is always a "but" though! In this case, the problem is that the splash window will appear in the list of active windows - which simply looks strange, and slightly unprofessional. (See gwb_fb2.app for an example.)

Shortly after this post, I was looking at photos of Clarion developers at Mike McLoughlin's Clarion Photo Gallery and while there, I came across a tip he had posted under his library of tips. The tip was described as "How to add a wallpaper/background image in CW2x" and was a simple explanation of using the Windows API function `EnableWindow`. This is a variation on the third option – you use the call to `EnableWindow` (conveniently renamed `DisableWindow`, with a suitable default parameter) to disable the background window, which also removes it from the open window list. Ten minutes later I had whipped up an application using this technique, and it was just what I wanted for setting my application frame background. (See gwb_fb3.app for an example.)

For most people this is probably an adequate solution, but I've always been a member of the "Lazy Programmers Club", and as I get older I find it harder to remember how to implement Windows APIs, or where to place the prototype for them, and I didn't want to go into the source code to edit my window etc. etc. so I thought this would be the perfect candidate for another kind of ten minute job. In other words, I thought this technique leant itself to a Clarion template that could be designed in 10 minutes, and then I'd never have to remember the actual code in the future. (So long as I could remember the name of the template!)

I'm reasonably conversant with the Clarion template language, so the task didn't appear too daunting. The first secret to writing a template is having a good idea of what it is you want it to do.

Template requirements

In this case the code required is simple, and my requirements for the template are straightforward.

1. I wanted the process to be implemented as a procedure template. That is, when I set the background frame on an application I want to do it as a new procedure, not as an extension, or code template, added to another procedure. This is because I want the technique to be simple to use and easy to find.
2. For similar reasons, I wanted to be able to design and edit the background frame window using the Clarion screen designer, called from the window properties button, rather than calling the designer from the source editor.
3. The original solution is implemented as a very basic source procedure; likewise, I wanted the template options and code to be as simple as possible.

Where to start?

The first problem I faced was that I had never designed a "Procedure" template before; nor a template that used the Clarion window designer. My usual starting point for a new template is to copy the code from an existing template, but I didn't even have a procedure template in my third party collection! Then I realized that the Clarion Splash procedure template must be pretty close to what I wanted. It opens a window, and has the window designer built in. But when I looked at that procedure I found that it has all the standard embeds of a Window procedure, which is overkill for this purpose. What I really wanted was a Source procedure template that had the window designer available, plus a few lines of source code built in.

So, I decided to start by copying the Clarion Source procedure template and spend the next nine minutes modifying it to match my needs. I designed the template using Clarion 5, but I copied the C6 ABC Source procedure template because it has some additional code that allows for the template to open and close any files referenced by the template - refer to comments above regarding the Lazy Programmers Club!

After renaming the Procedure template, using a convention of placing my initials "GWB" wherever I can so I can find my code when I need to, and placing it within a template (.TPL) file, it was ready to go. It just didn't do anything useful yet. The rest of this article describes the steps need to produce the final template.

Adding the Window Designer

Adding the Window Designer to the procedure proved to be dead easy. I just added the Word `WINDOW` to the `#PROCEDURE` prototype:

```
#PROCEDURE(GWBFBS,'GWB Frame Background Source'),WINDOW
```

I added the template to an application, and indeed the window designer was there; inside a source procedure! I designed my window with glee! Unfortunately, when I saved the procedure, and looked at the source code, my window definition wasn't there. I had assumed the Clarion Window designer would automatically generate the appropriate code. I had forgotten that the template language specifies pretty much everything Clarion generates.

I had a look at the template code in the Splash procedure to see if I could find where it generated the code for the Window structure, but couldn't find it. Then I looked at the declaration of the Splash procedure template.

```
#PROCEDURE(Splash,’Splash Window’),WINDOW,HLP(‘~TPLProcSplash’)
       ,PARENT(Window(ABC))
```

That bit on the end looked promising; `PARENT(Window(ABC))`. I added that to my template declaration, re-opened my test application and now I also had the Window code in my source; unfortunately, all the `#EMBED`s for a Window procedure had also been added. In other words, I had converted my Source template to a Window template simply by adding that one statement. That's not what I wanted, so I took a step backwards and went looking for the template code that actually generated the Window code. Not surprisingly it was in the `Window(ABC)` template, the parent template for the Splash template.

Finding the section of a template that does a specific task can sometimes be difficult, and daunting to someone who is not familiar with the template language. In this case I was able to find the code I needed by locating the embed points before and after the window structure.

```
#EMBED(%DataSectionBeforeWindow,'Data Section, Before Window Declaration')
        ,DATA,LEGACY #IF(%WindowStatement)
#IF(SUB(%WindowStatement,1,11)='APPLICATION')
120 odd lines of other stuff not worth printing here
#ENDIF
#EMBED(%DataSectionAfterWindow,'Data Section, After Window Declaration')
```

```
          ,DATA,LEGACY
```

I simply cut and pasted this (*not* omitting the 120 odd lines) to my template, and it worked, without all the excess baggage of the window procedure embed points. The reason this works is because the Clarion window designer saves everything specified in a window to a template "symbol" (variable) called `%WindowStatement`.

If `%WindowStatement` is used explicitly in a template it will generate only one line of code, the `Window` declaration, for example.

```
WINDOW('Browse the States File'),AT(,,255,220)|
         ,FONT('MS Sans Serif',8,,),IMM,HLP('BrowseStates')|
                 ,SYSTEM,GRAY,RESIZE,MDI
```

Everything else in a window design is also saved in `%WindowStatement`, but has to be extracted from this symbol, literally by using the `EXTRACT` statement. That's what those 120 odd lines of code do.

Tidying up the window

If you had a play around with gwb_fb3.app above, you might have found that sometimes things can go wrong. For instance, what happens if I add a title, system menu, drop menu or toolbar to the window? Should the window be SDI or MDI? How big should the window be? What if I forget to set its position properly? These are things I don't want to worry about next time I use this template, so I'll use the template to make sure my frame background window has all the appropriate run-time specifications. The template section described above generates "window" code exactly as I designed the window. I decided to modify this template code so that it generates window code as it needs to be for the frame background procedure. This will prevent me from making mistakes at design time. Next week I'll show you how to modify that template code.

Download the source

Reader Comments

Add a comment

Clarion Magazine

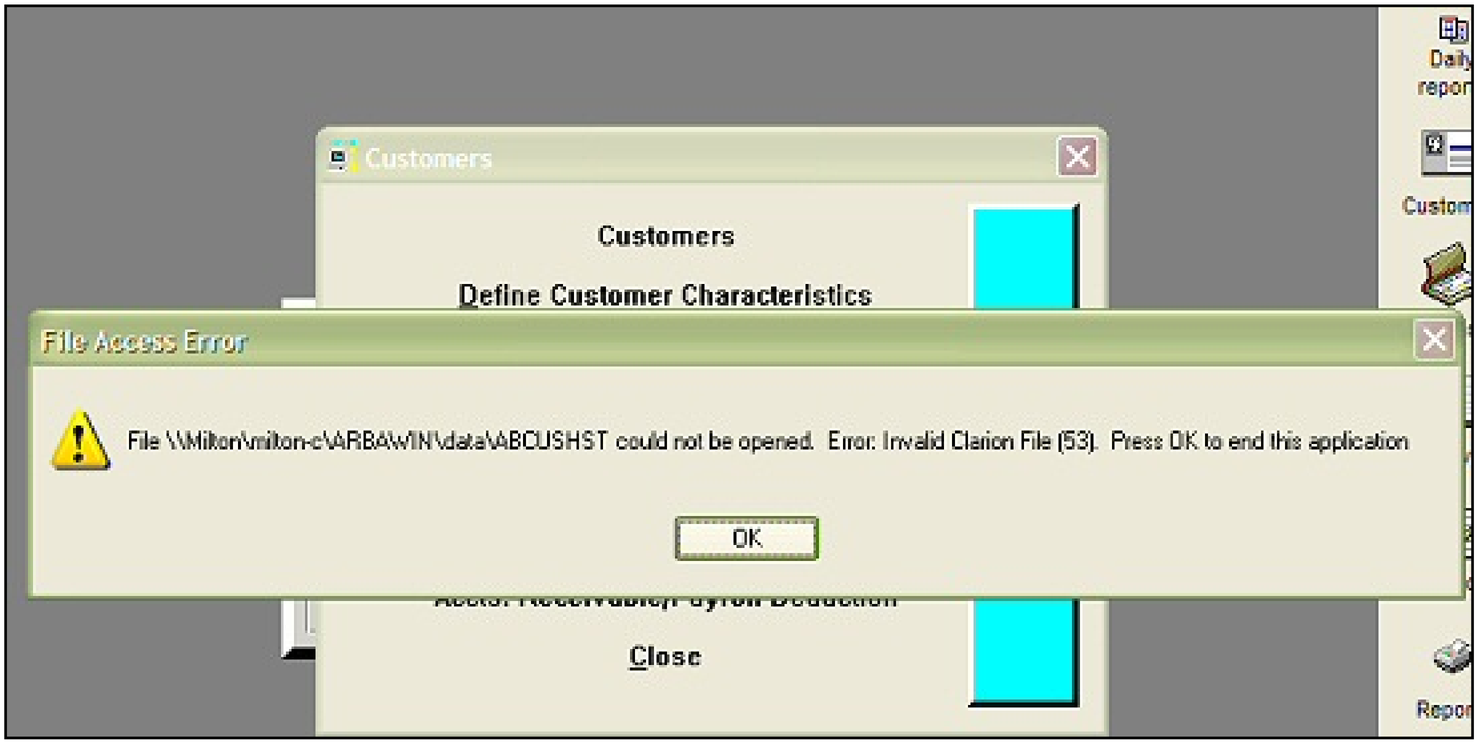
Hello, Server?

by Henry Plotkin

Published 2005-12-12

If my application runs across a network, I want to know if that network is working or not. If the network drops while my program is trying to access files, my user might get an error message. Or they might not. Or they might get a "Save As" dialog, as happened recently.

If they do get a message, it is unlikely that they will comprehend it. For example, trying to open a browse after unplugging the network cable, this message appeared:



File \\Milton\milton-c\ARBAWIN\data\ABCUSHST could not be opened. Error: Invalid Clarion File (53). Press OK to end this application

Figure 1. Message for downed network (full sized image)

Without consulting the manuals, I am not sure what error code 53 is. After consulting the manuals, I still do not know what it is. This error is not listed in "Trappable Runtime Errors." My users are certainly not going to know what this means. My users will not know what to do about it either. My users will certainly call me.

If no message is presented (I have seen a no message condition when trying to save a Microsoft Word document across a network after unplugging the cable) or a "Save As" dialog pops up, my users think ill of my software. None of these circumstances is very good for me.

It would be useful for the program to discover the outage before such an error message or instead of no message. On detecting a network outage, the program could present a message more easily understood by the typical user

Searching the MSDN library eventually revealed two API calls that appear to address the need:

IsNetworkAlive

and

IsDestinationReachable

Further reading demonstrates that `IsNetworkAlive` is not suitable.

> The **IsNetworkAlive** function determines whether or not a local system is connected to a network, and identifies the type of network connection, for example, a LAN, WAN, or both.

This means only that `IsNetworkAlive` will tell whether there is a network connection and, if so, what type of connection (LAN, WAN, etc.). It will not tell whether the server containing the data is "alive."

`IsDestinationReachable`, however, will give the essential information:

> The **IsDestinationReachable** function determines whether or not a specified destination can be reached, and provides Quality of Connection (QOC) information for a destination.

Checking the documentation for `IsDestinationReachable` indicates some important restrictions, conditions and quid pro quos.

The first is that this function requires Internet Explorer 5 or higher, regardless of operating system. IE 5 installs SENSAPI.DLL which contains this function in two versions (ANSI and Unicode). This is revealed when using LibMaker to create a Clarion .LIB for inclusion.

The demonstration app, downloadable at the end of this article, contains the required Clarion .LIB created with LibMaker. You now only need to include it in your project ("Library, object, and resource files"). If you do not have Clarion 6, a locally compiled version is included for your testing.

The `IsDestinationReachable` call takes two parameters.

First is a pointer to a null terminated string. This string contains the destination to check. In Clarion, this becomes `*CString`.

Also, testing demonstrates that final backslashes should not be included. The same parameter that succeeds without a final backslash, fails with it.

The second parameter is a pointer to "Quality of Connection information." Checking the link to QOC info shows that this is a structure. Therefore, in Clarion, the address of a group is required.

This group is declared as:

```
QOCInfo                GROUP,PRE()
dwSize                   LONG
dwFlags                  LONG
dwInSpeed                LONG
dwOutSpeed               LONG
                       END
```

(It seems foolish, does it not, to have to include the size of the group within the group. Perhaps the creator of this call considers that I can compute this but my compiler cannot!)

The final prototype for the `IsDestinationReachable` API therefore is:

```
Module('sensapi.lib')
```

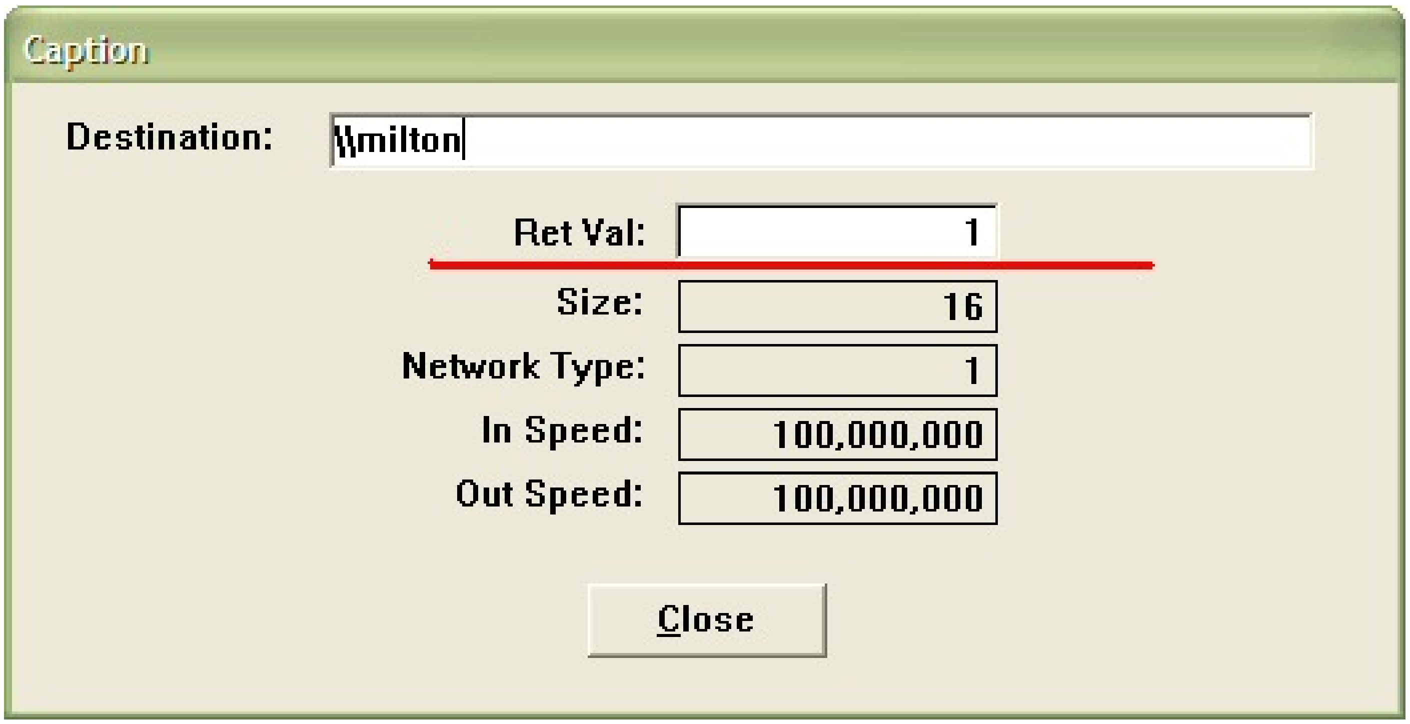
```
      IsDestinationReachable(*CString lpszDestination|
        ,LonglpQOCInfo),Name('IsDestinationReachableA')|
        ,Long,raw,pascal
    End
```

Before implementing `IsDestinationReachable`, attend to the documentation of the `lpszDestination` parameter:

> A pointer to a null-terminated string that specifies a destination. The destination can be an IP address, UNC name, or URL.

Only an IP address, UNC name or URL may be passed. The first specific consequence is that `IsDestinationReachable` returns information on servers, not on resources on servers.

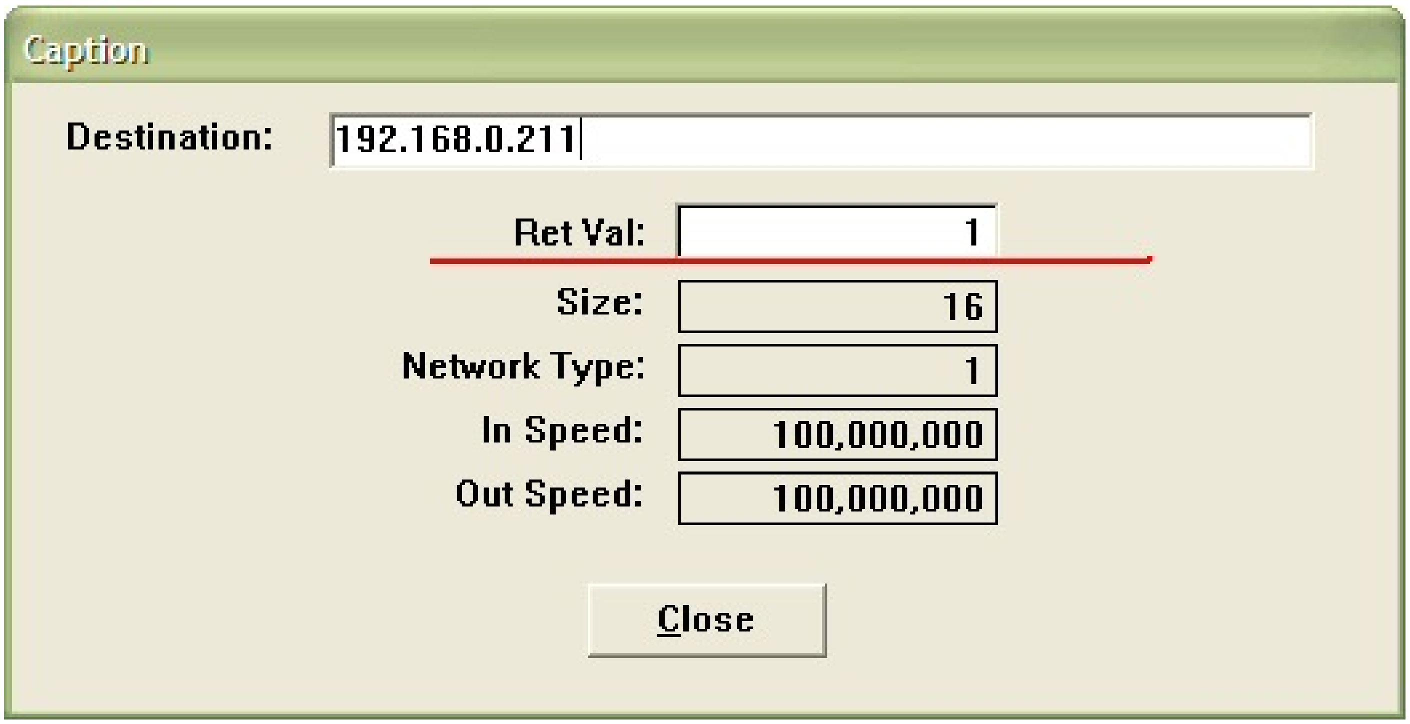
Passing the UNC name of another computer on the network returns the anticipated result:



Destination:	\\milton
Ret Val:	1
Size:	16
Network Type:	1
In Speed:	100,000,000
Out Speed:	100,000,000

Figure 2. Results for UNC name

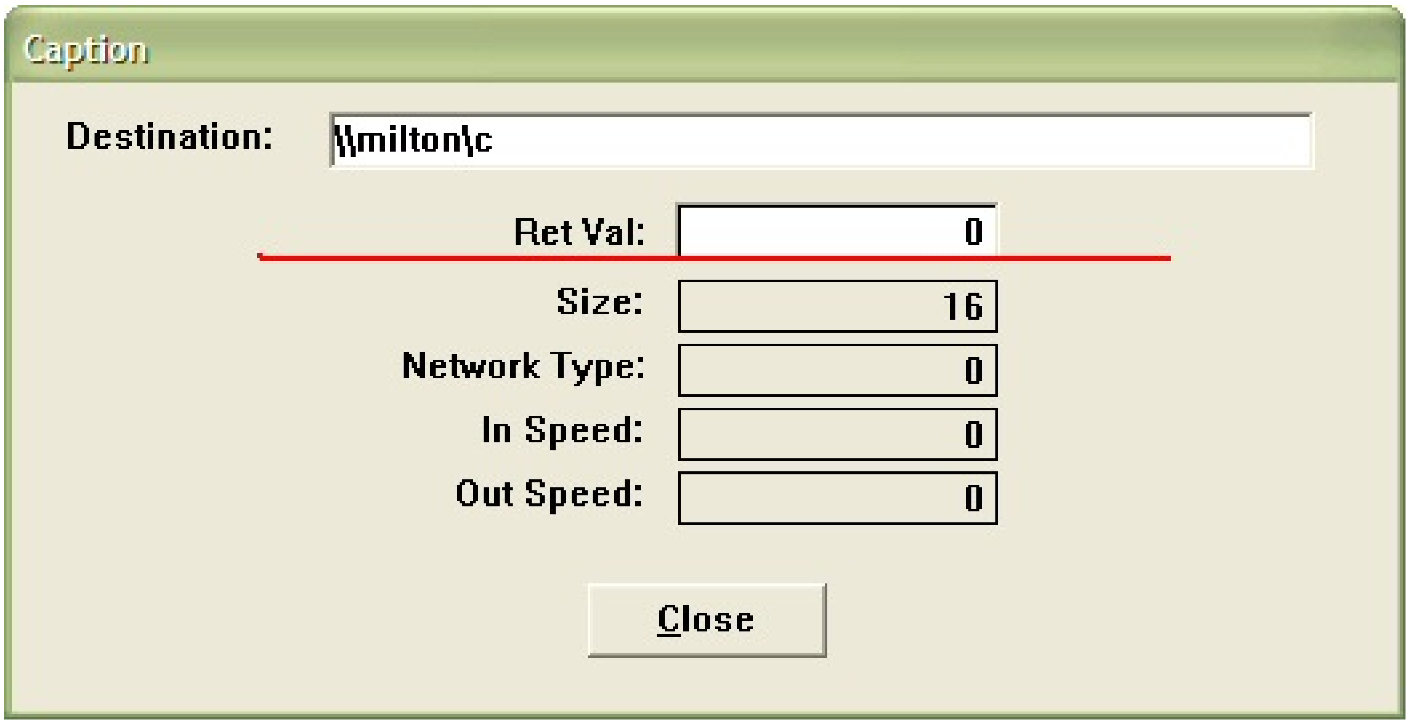
Passing the IP address of another computer on the network returns the anticipated result:



Destination:	192.168.0.211
Ret Val:	1
Size:	16
Network Type:	1
In Speed:	100,000,000
Out Speed:	100,000,000

Figure 3. Results for IP address

Passing the UNC name including a drive fails:

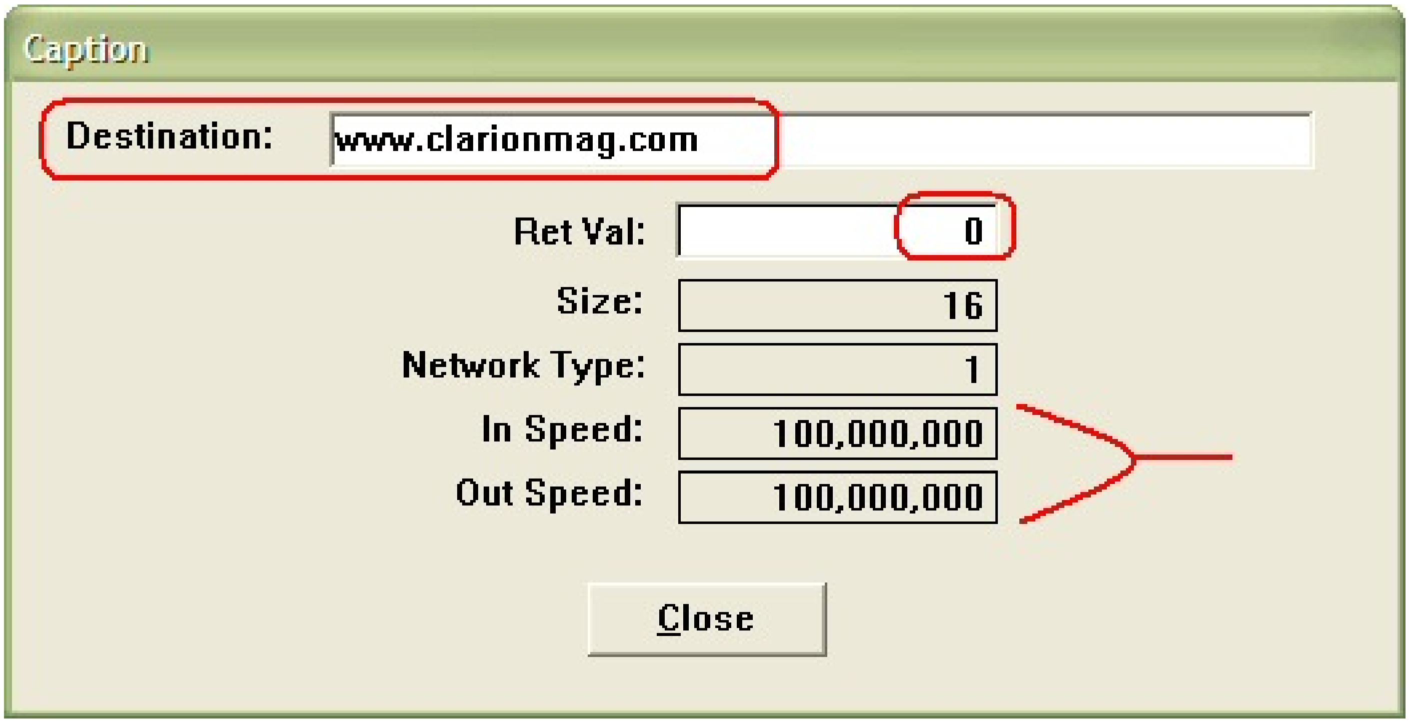


Destination:	\\milton\c
Ret Val:	0
Size:	16
Network Type:	0
In Speed:	0
Out Speed:	0

Figure 4. UNC name with drive

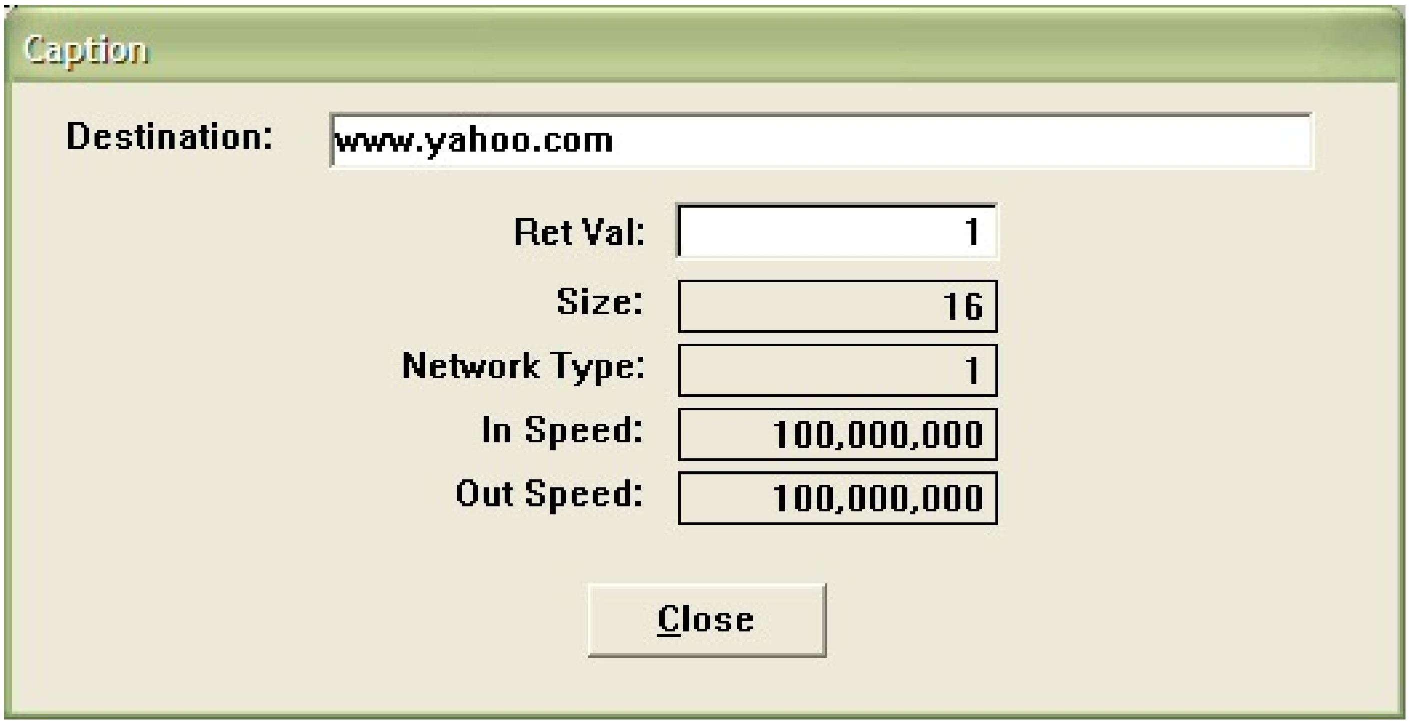
The next restriction to note is that `IsDestinationReachable` is available only for computers connected through TCP/IP.

I also note that URLs can be unreliable. Figure 5 shows no connection, but a non-zero connection speeed.



Destination:	www.clarionmag.com
Ret Val:	0
Size:	16
Network Type:	1
In Speed:	100,000,000
Out Speed:	100,000,000

Figure 5. Apparent failure using URL (the web site *is* accessible)



Destination:	www.yahoo.com
Ret Val:	1
Size:	16
Network Type:	1
In Speed:	100,000,000
Out Speed:	100,000,000

Figure 6. Expected result using URL

Another version of the demonstration program tests `GetLastError`. Whenever

`IsDestinationReachable` returns `False`, `GetLastError` always returns `1232`. `GetLastError` returns `1232` for some good URLs, as in Figure 6, as well as made up IP addresses.

This error means "The network location cannot be reached. For information about network troubleshooting, see Windows Help" according to MSDN. Most enlightening.

For this reason, Microsoft's statement "You can supply a NULL pointer if you do not want to receive the QOC information" is not a good idea I think. (In the demonstration app, you may comment out the API call which includes the pointer and substitute the call which uses a null pointer for testing.)

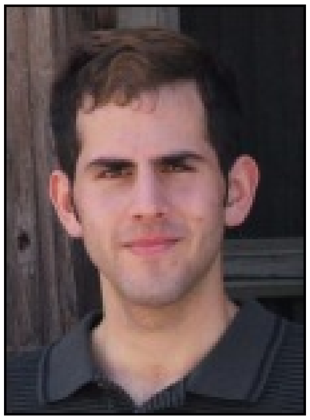
Because `IsDestinationReachable` can fail when testing good URLs, I would test both `RetVal` and `dwInSpeed` (or `dwOutSpeed`). If either one is non-zero, the connection test succeeds. If both are zero, there is a disconnect:

```
If RetVal = 0 and dwInSpeed = 0
  ! error message
  ! action
End
```

Now I know the network died before users call me, and my program can act to prevent damage that might otherwise ensue.

Download the source

"hp" in fact prefers Hewlett-Packard printers but will use whatever is available. Born in New York City, hp is a self-taught Clarion developer doing a substantial amount of work for hospital gift shops.



Reader Comments

Add a comment

- » Nice article Henry. With regard to : >> (It seems...

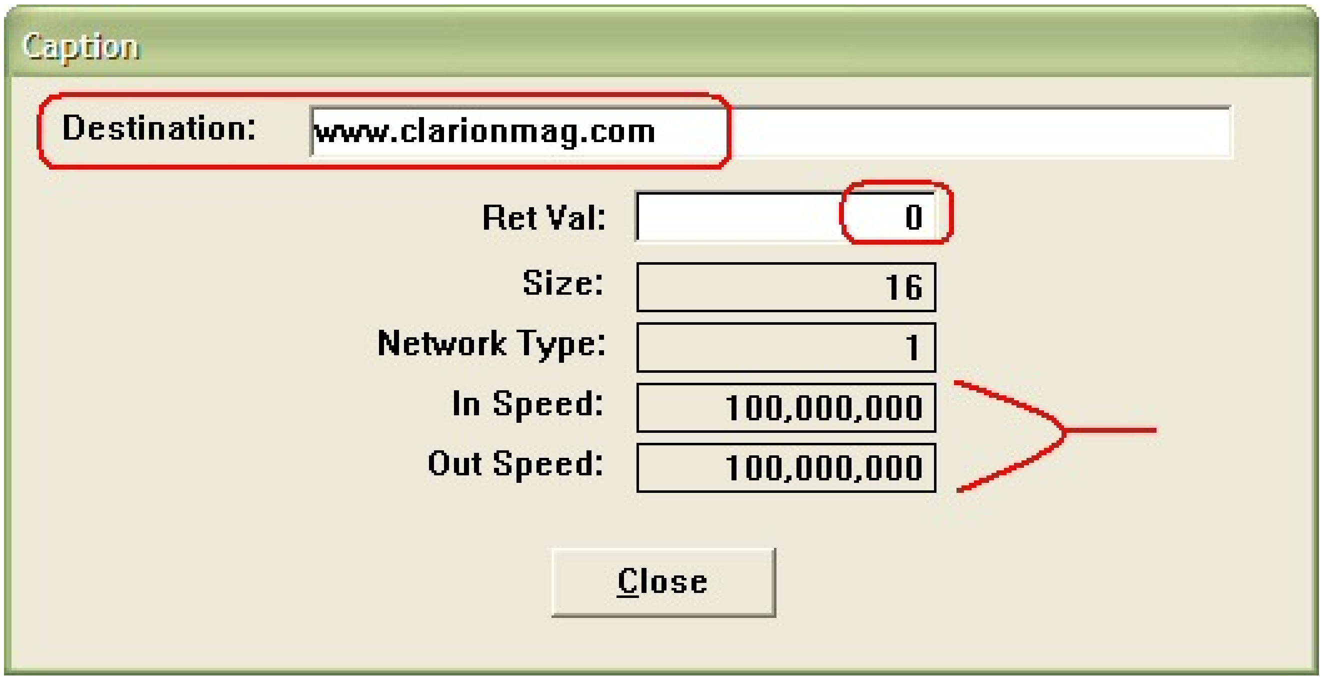
Clarion Magazine

Using GetLastError and FormatMessage to Check the Network

by Henry Plotkin

Published 2005-12-15

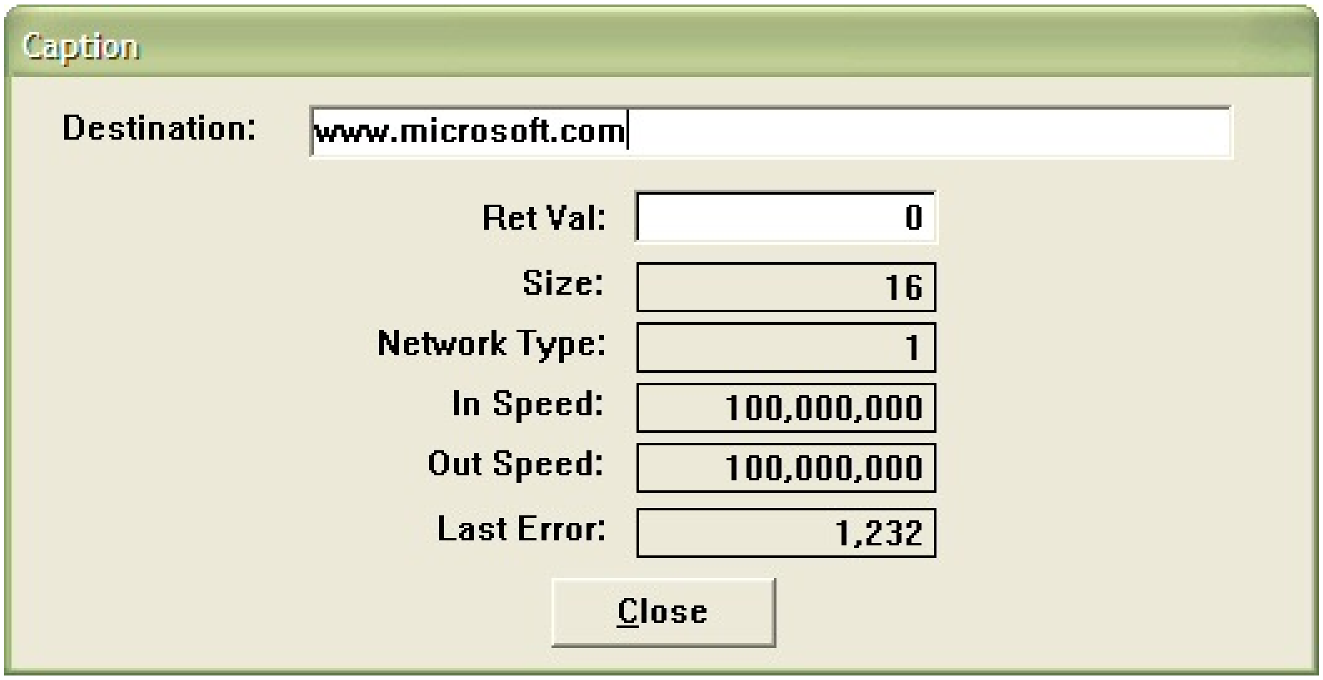
My discussion of the `IsDestinationReachable` API call made, I think, a bad implication. In that article, I said that checking by URL was not reliable. As an example I showed the results of checking for www.clarionmag.com:



Destination:	www.clarionmag.com
Ret Val:	0
Size:	16
Network Type:	1
In Speed:	100,000,000
Out Speed:	100,000,000

Figure 1. Is www.clarionmag.com reachable? Yes, but IsDestinationUnreachable doesn't think so.

To correct any possible misimpression, note that Clarion Magazine seems not to be alone:



Destination: www.microsoft.com

Ret Val: 0

Size: 16

Network Type: 1

In Speed: 100,000,000

Out Speed: 100,000,000

Last Error: 1,232

Figure 2. Another "unreachable" site

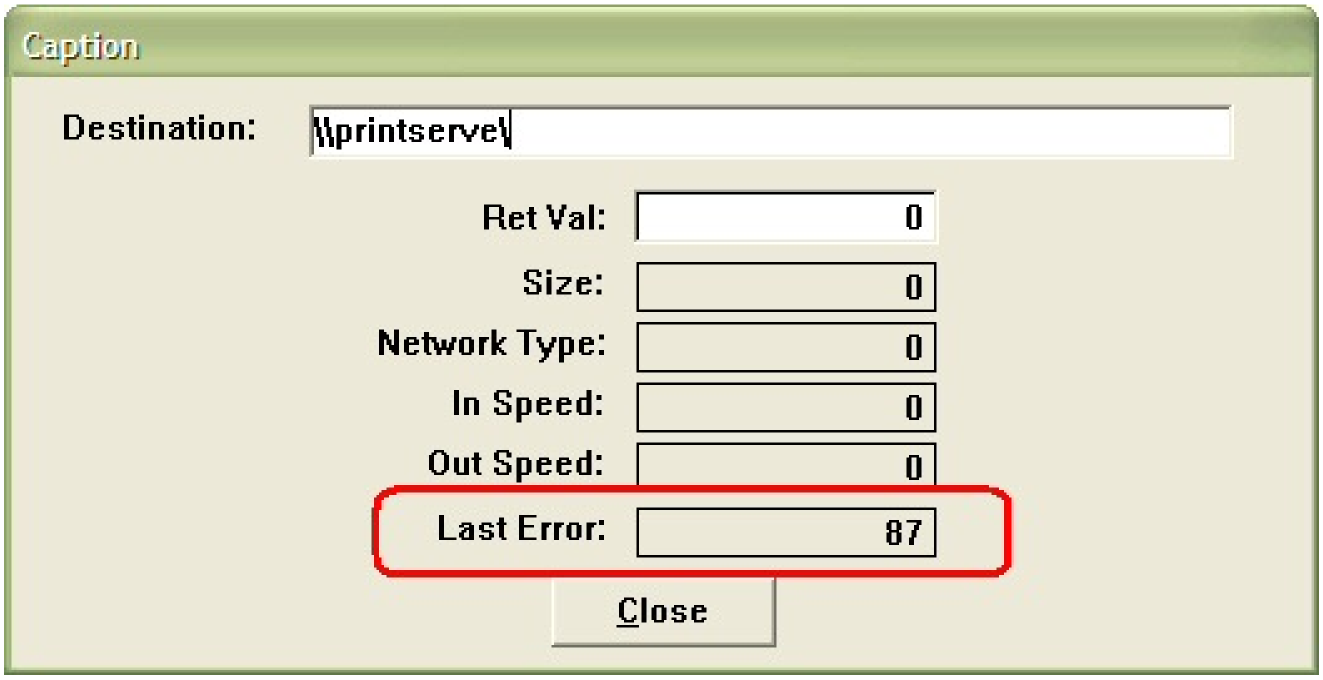
Whether this means that Clarion Magazine is or is not in good company, I cannot say. I must not comment, I think, on the unreachability of Microsoft.

It is possible, I am informed, that `IsDestinationReachable` uses a PING. Clarion Magazine, I am informed, does not permit PINGing. Probably, Microsoft doesn't either. If so, this reinforces my thinking that URLs cannot be relied on with `IsDestinationReachable`'s return value. One must also check the `QOCInfo` group to see if the in or out speed is non-zero.

> **Editor's note:** Henry is correct – the Clarion Magazine firewall is perhaps overly paranoid, and doesn't allow much besides HTTP and SSH. As noted on MSDN, `IsDestinationReachable` "does not work through HTTP proxies or firewalls that restrict ICMP ping packets."

I also said in that article that whenever `IsDestinationReachable` returns `False`, `GetLastError` always returns `1232`. This is not true.

I said that I had another version of my network checking demonstration that implemented `GetLastError` (this version, the subject of this article, can be downloaded at the end of this article; there is a locally compiled EXE included if you do not use C6). After finishing the last article, I decided to complete the implementation of `GetLastError`. I discovered that other error codes were possible:



Destination:	\\printserve\
Ret Val:	0
Size:	0
Network Type:	0
In Speed:	0
Out Speed:	0
Last Error:	87

Figure 3. Error other than 1232

This proves what I said about final slashes not working correctly. The same error occurs when appending a drive letter (e.g. `\\printserve\c`).

This brings today's problem. When `IsDestinationReachable` errors, I know there is an error. `GetLastError` tells me what the error code is. But I do not know what the error code means.

When Clarion throws an ErrorCode 47, I know what it means. If I get a message with an ErrorCode 2 or 35 or 37, I know what it means. Microsoft's errors mean nothing to me.

But Microsoft has many, many errors. Printed out, Microsoft's error list sits almost 15mm thick (82 pages when I printed it two years ago; surely new errors have since been introduced).

In the previous article, I followed the link to "System Error Codes" from the `GetLastError` page and found that error code 1232 meant:

> The network location could not be reached. For information about network troubleshooting see Windows Help.

I do not find this very informative. But my users may find this better than some of the other errors that can happen (no message, save as, ErrorCode 53 – whatever that is). Of course, if it stops them from calling me needlessly, it is wonderfully descriptive.

So I want Microsoft to explain its errors. I explain mine, they should explain theirs. This is fair.

The documentation for `GetLastError` states "To obtain an error string for system error codes, use the `FormatMessage` function. For a complete list of error codes provided by the operating system, see `System Error Codes`." (This link to System Error Codes is to the English version. For other languages, follow the link on `GetLastError`.) One presumes that "an error string" is a description that a user might comprehend.

So, to implement a more user friendly messaging system for network errors, both `GetLastError` and `FormatMessage` must be implemented.

GetLastError

`GetLastError` returns the last *thread-specific* error from the O/S. This means that anything happening in the code after the point at which an error might happen could change the error code.

Microsoft concurs in this, stating:

> You should call the **GetLastError** function immediately when a function's return value indicates that such a call will return useful data.

To me, this means that I will not simply want to check `GetLastError` but also to store the return value, using the stored value for my work and allowing the O/S to throw more errors. Then, whatever may happen after, I have a testable variable.

Microsoft prototypes `GetLastError` as:

```
DWORD GetLastError(void);
```

`GetLastError` takes no parameters, returning a `DWORD`.

In Clarion, this becomes:

```
GetLastError(),LONG,PASCAL
```

(Actually a `DWORD` is a `ULONG` but a `LONG` will do here – the highest error number listed on MSDN will still fit in a long. Perhaps they are planning for the future.)

To actually use `GetLastError`:

```
Err = GetLastError()
```

(where `Err` is my variable).

FormatMessage

Once a useable value is returned, "To obtain an error string for system error codes, use the `FormatMessage` function."

There is an example of using `FormatMessage` in the Microsoft article "Retrieving the Last-Error Code." If you can follow the C code in this example, do so, for the example makes it clear that actual implementation may be easier than the documentation implies.

`FormatMessage` takes seven parameters:

```
DWORD    dwFlags
LPCVOID  lpSource
DWORD    dwMessageId
DWORD    dwLanguageId
LPTSTR   lpBuffer
DWORD    nSize
va_list* Arguments
```

Only the third and fifth are really important. Parameter three is the error code from `GetLastError`. Parameter five is a `CString` where the resulting error message goes and which the programmer can use. The remaining parameters are mostly confusing.

The prototype for `FormatMessage`, then, is:

```
FormatMessage(ULong,ULong,Ulong,ULong,*CString,ULong,Long), |
  ULong,Pascal,raw,Name('FormatMessageA'),PROC
```

> `Note: Name('FormatMessageA')` indicates I am using the ANSI version, not the Unicode version. The `PROC` attribute means that I can call `FormatMessage` as a simple statement, even though it returns a value, without a compiler warning.

So, in the Global Map, place the following to ensure that both required APIs are available:

```
MODULE('win32.lib')
  GetLastError(),LONG,PASCAL
  FormatMessage(ULong,ULong,Ulong,ULong,*CString,ULong,Long),+
    ULong,Pascal,raw,Name('FormatMessageA'),PROC
 END
```

The Parameters

`dwFlags`: Without doubt, this is the most confusing parameter. With `dwFlags`, the API is told how the output is to be formatted. Much here is not relevant in a Clarion program. Most important is that I infer that I need equates for the various values that may be passed:

In the Global, Before File Declarations or Before/After GLOBAL Includes embed:

```
FORMAT_MESSAGE_ALLOCATE_BUFFER EQUATE(00000100H)
FORMAT_MESSAGE_ARGUMENT_ARRAY  EQUATE(00002000H)
FORMAT_MESSAGE_FROM_HMODULE    EQUATE(00000800H)
FORMAT_MESSAGE_FROM_STRING     EQUATE(00000400H)
FORMAT_MESSAGE_FROM_SYSTEM     EQUATE(00001000H)
FORMAT_MESSAGE_IGNORE_INSERTS  EQUATE(00000200H)
FORMAT_MESSAGE_MAX_WIDTH_MASK  EQUATE(000000FFH)
va_list                        EQUATE(LONG)
```

(While suggested by the MSDN documentation, this specific list is borrowed from a FAQ by Mr. Stephen Bottomley found in Mr. Steven Parker's Just theFAQs knowledge base.)

`FORMAT_MESSAGE_ALLOCATE_BUFFER` means the API should allocate memory for the buffer. Because I am specifying my own variable to receive output, this is unnecessary.

`FORMAT_MESSAGE_IGNORE_INSERTS` is used to declare whether or not additional characters (created in the Arguments parameter) are to be ignored. I will not be inserting anything into the error description. So this flag is desired.

`FORMAT_MESSAGE_FROM_STRING` seems to be involved with inserted characters. I do not want this option.

`FORMAT_MESSAGE_FROM_HMODULE` points to a module containing the list of error messages. I do not want this option. I want to use system-provided messages.

`FORMAT_MESSAGE_FROM_SYSTEM` tells the API to use the system error code list. I want this flag set (and, therefore, the message will appear in the correct language, too).

`FORMAT_MESSAGE_ARGUMENT_ARRAY` is irrelevant. Because I am setting `FORMAT_MESSAGE_IGNORE_INSERTS`, this parameter is ignored.

The **`lpSource`** parameter is ignored because neither of the flags required in the documentation will be set. Zero will do here.

dwMessageId is the error code returned by `GetLastError`.

dwLanguageId is zero because the message text is coming from the PC's own operating system.

lpBuffer is the label of a `CString` in which the message will be stored and which can be displayed to the user.

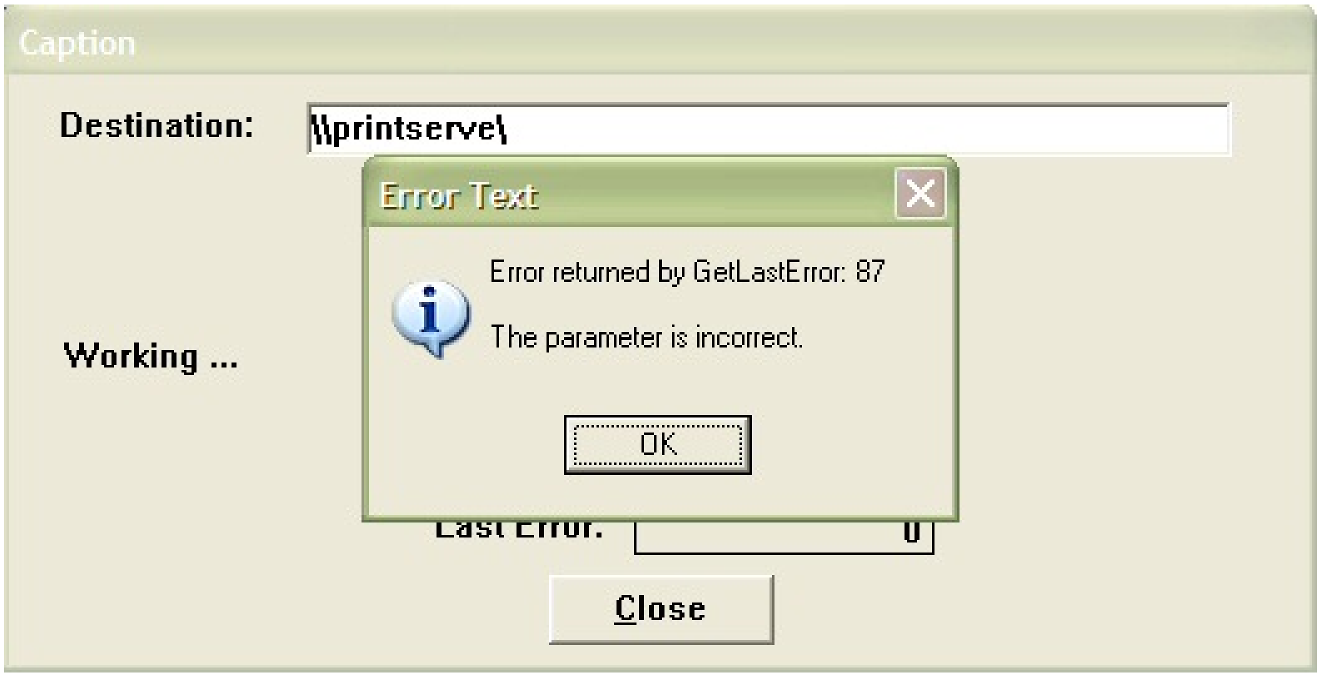
nSize because, of course, I can count the size of the `CString` better than the compiler, I must tell the API the size of the string to be used.

The final parameter list looks like this:

```
FormatMessage(      |
FORMAT_MESSAGE_FROM_SYSTEM+FORMAT_MESSAGE_IGNORE_INSERTS,  |
   0,|
   Err,|
   0,|ErrorText,|
   size(ErrorText),|
   0)
```

The final call and use of `FormatMessage` looks like:

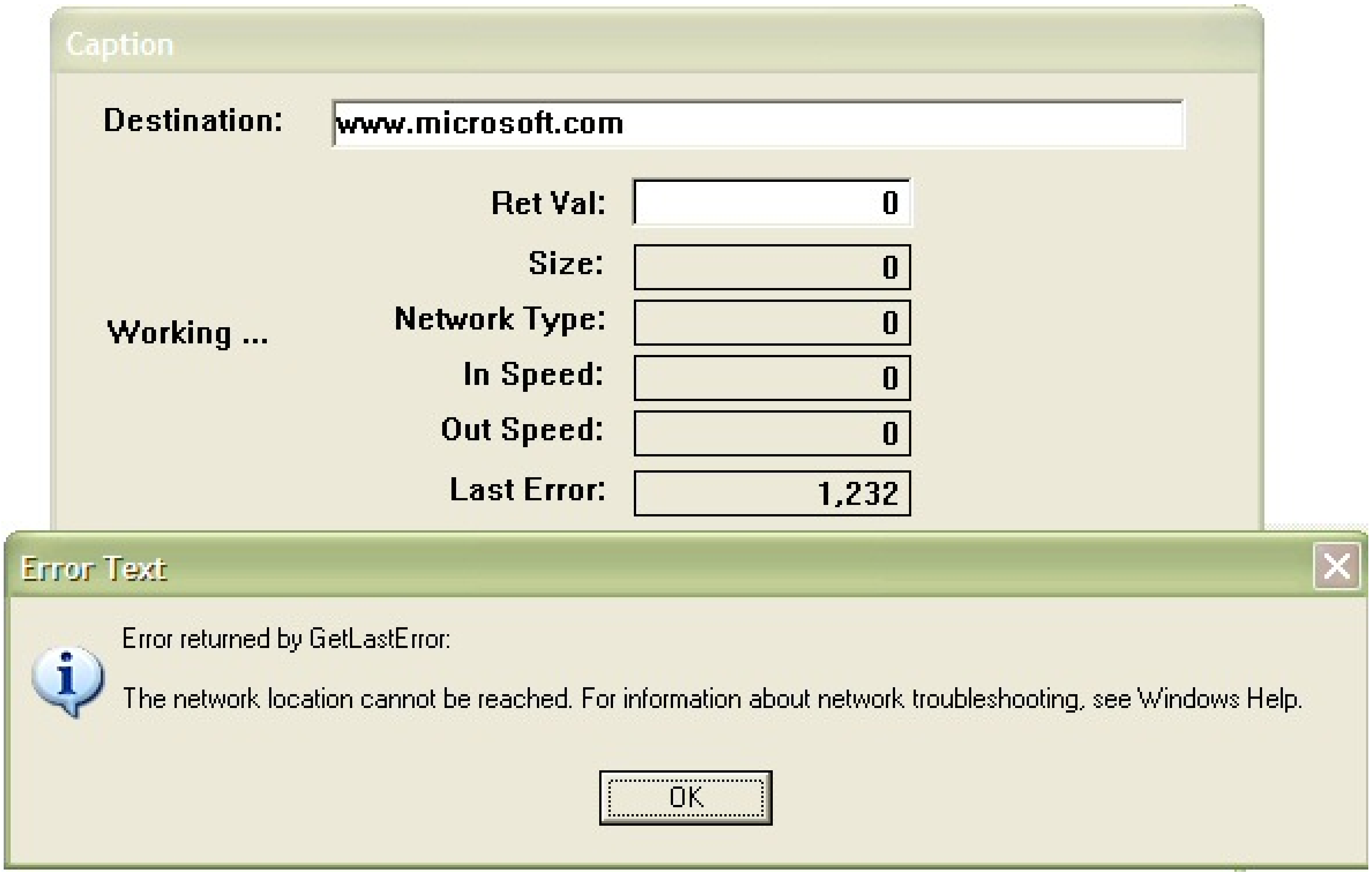
```
Err = GetLastError()
If Err <> 0
  FormatMessage(FORMAT_MESSAGE_FROM_SYSTEM+  +
  FORMAT_MESSAGE_IGNORE_INSERTS,0,        +
  Err,0,ErrorText,size(ErrorText),0)
  Message('Error returned by GetLastError: <13,10>{2}' & +
           ErrorText,'Error Text',Icon:Asterisk)
End
```



Error returned by GetLastError: 87

The parameter is incorrect.

Figure 4: FormatMessage in operation



Error returned by GetLastError:

The network location cannot be reached. For information about network troubleshooting, see Windows Help.

Figure 5: Another example

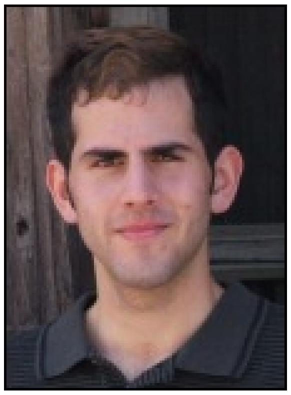
Of course, one must not forget to declare a `CString` to pass to the API.

Summary

It was my goal, originally, to change the user's experience when a network went down. `IsDestinationReachable` allows me to capture the network failure in a timely manner. `GetLastError` and `FormatMessage` allow me to present messages, readable messages, to the user, informing them of the problem. All in all, much better than cryptic messages like the Errorcode 53 message or, worse, no message at all.

Download the source

"hp" in fact prefers Hewlett-Packard printers but will use whatever is available. Born in New York City, hp is a self-taught Clarion developer doing a substantial amount of work for hospital gift shops.



Reader Comments

Add a comment

Clarion Magazine

Modifying The Frame Background - A 10 Minute Template, Part 2

by Geoff Bomford

Published 2005-12-16

Last week I introduced the idea of a window that functions as an app frame background, and showed the source code (which I got from Mike McLoughlin's Sterling Data site. I also began the creation of a "ten minute" procedure template to automate background creation for future apps. In this second part of the article I'll continue with the modifications to the template code.

The first step is to replace the window statement with a hard-coded window specification. Originally I was going to set the window size at 1600 * 1200, since this will fill the background on most frames:

```
WINDOW,AT(0,0,1600,1200)
```

This specifies a window in the top left corner of the frame, 1600 dialog units wide and 1200 high. After some experimentation with the template I thought it was better to use whatever size window the designer specified, and do this by extracting the information from the window definition. Then I extract all the values that are safe to use for the frame background, namely `FONT`, `COLOR`, `WALLPAPER`, `TILED`, `CENTERED` and `TIMER`. These are windows properties I can modify at design time by specifying the font, color and background image, if any, and timer attribute. I can use these values to build a new `Window` code statement, and append `MDI` to the statement, since the window always has to be MDI. So the first line of my window statement code might start out looking like this:

```
BackWindow WINDOW('Back frame'),AT(0,0,432,227)|
   ,FONT('Comic Sans MS',8,,FONT:regular,CHARSET:ANSI)|
   ,COLOR(0FFFF80H), IMM,ICON('MEMBER.ICO'),ALRT(MouseRight)|
   ,STATUS,WALLPAPER('frame1.gif'), TILED,TIMER(100),SYSTEM|
   ,TOOLBOX,GRAY,MAX,MAXIMIZE,DOUBLE,MASK,SDI,MODAL
```

…but my template transforms it to this:

```
BackWindow           WINDOW,AT(0,0,432,227)|
   ,FONT('Comic Sans MS',8,,FONT:regular,CHARSET:ANSI)|
   ,COLOR(0FFFF80H),WALLPAPER('frame1.gif'),TILED,TIMER(100),MDI
```

From the template you can see that I looked for specific keywords to allow in my background window, using the following syntax

```
#IF(EXTRACT(%WindowStatement,'FONT'))
    #SET(%GWBWindowStatement,%GWBWindowStatement & ',' &
EXTRACT(%WindowStatement,'FONT'))
  #ENDIF
```

Similarly I included 'COLOR', 'WALLPAPER', 'TILED', 'CENTER' and 'TIMER'. If TIMER was present I also set a %GWBTimerPresent symbol to "1", so I know to include an embed for Timer events in the template…

```
        #SET(%GWBWindowStatement,%NewWindowStatement & ',' &
    EXTRACT(%WindowStatement,'TIMER'))
        #SET(%GWBTimerPresent,1)
```

The line in the template which generates the Window statement in the source code is…

```
    %[20]Window %GWBWindowStatement
```

The %[20]Window is template code which means "place the contents of the %Window symbol here, then leave 20 spaces before generating the symbol contents", and it generates the code shown above.

Now, what about anything the designer has placed on the window? Here I want the template to do the same thing, keep what is safe to use and remove anything that wouldn't be appropriate. So the next section of the template loops through the %Control "objects" placed on the window and selectively keeps or removes them, depending on their %ControlType. Any control type before the following #ELSE statement will be eliminated…

```
        #CASE(%ControlType)
        #OF('MENU')
        #OROF('MENUBAR')
        #OROF('TOOLBAR')
        #OROF('BUTTON')
        #OROF('ITEM')
        #OF('OPTION')
        #OROF('GROUP')
        #OROF('SHEET')
        #OROF('TAB')
        #OROF('OLE')
    %[22+(2*%GWBIndentation)]Null %GWBControlSourceLine
          #SET(%GWBIndentation,%GWBIndentation+1)
        #ELSE
    %[22+(2*%GWBIndentation)]Null %GWBControlSourceLine
        #ENDCASE
      #LOOP,WHILE(%GWBIndentation)
        #SET(%GWBIndentation,%GWBIndentation-1)
    %[22+(2*%GWBIndentation)]Null END
      #ENDLOOP
    %[20]Null END
    #EMBED(%DataSectionAfterWindow,'Data Section, After Window Declaration'),DATA,LEGACY
    #?
    #RESUME
    #ENDAT
```

%GWBIndentation is a symbol used to keep track of the level of indentation and also how many END statements are needed, and where they should be placed. Options, Groups, Sheets, Tabs and OLEs all need an END statement so the above code increments and decrements %GWBIndentation appropriately.

Combined with the %[Number] syntax, described above, %GWBIndentation ensures the code is correctly formatted and produces the final window code section. Here's the code from gwb_fb4.app:

```
    ! Window Structure
    BackWindow  WINDOW,AT(0,0,640,480)|
                  ,FONT('Comic Sans MS',8,,FONT:regular,CHARSET:ANSI)|
                  ,COLOR(COLOR:Blue),TILED,MDI
                  STRING('This background widow contains a STRING, PROMPT, ' &|
                                'PANEL and IMAGE control.'),AT(47,41),USE(?String2)|
                                    ,TRN,CENTER,FONT(,,080FFFFH,,CHARSET:ANSI)
                  PROMPT('This is a Prompt control that can be used to ' &|
                    'display multiple lines of text, up to 256 characters!')|
                                ,AT(87,71,159,58),USE(?Prompt1),CENTER|
                                ,FONT(,14,COLOR:White,,CHARSET:ANSI)
                  PANEL,AT(63,156,206,37),USE(?Panel1),FILL(COLOR:White)
                  IMAGE('hdidt.gif'),AT(71,161),USE(?Image1)
                END
```

Just as an aside, I think you can see from the above technique how it might be possible to take a standard Clarion window structure and generate a window structure for a Java, .NET or perhaps any other language. I'm not saying it would be easy, but it's certainly possible.

```
#DEFAULTs
```

That takes care of my actual `Window` structure, but there is an additional feature of Window templates that can be very handy here as well, the `#DEFAULT` section. `#DEFAULT` is a template code section where I can design a default window to be used by my `WINDOW` template. In this template the code is as follows:

```
#DEFAULT
NAME GWBFrameBackgroundSourceProcedure
[COMMON]
DESCRIPTION 'Frame Background Source Procedure'
FROM GWBFBW GWBFBS
[WINDOW]
BackWindow WINDOW,AT(0,0,640,480),FONT('Comic Sans MS',8,|
             ,FONT:regular,CHARSET:ANSI),COLOR(COLOR:White), |
             TILED,MAXIMIZE,DOUBLE,MDI
             STRING('This is my Application!'),AT(41,41,182,10)|
                          ,USE(?String2),CENTER,#ORIG(?String2)
             PANEL,AT(10,71,267,143),FILL(COLOR:Silver),BEVEL(-2,3)
             STRING('Created with Clarion!'),AT(47,138,182,10)|
                          ,USE(?String1),CENTER,#ORIG(?String1)
           END
#ENDDEFAULT
```

The only tricky bit to this is getting the `FROM` parameters correct: there are two, the `#TEMPLATE` name, and the `#PROCEDURE` name. For this template I used `GWBFBW` and `GWBFBS` respectively. I deliberately kept these names short because they appear in the procedure tree, and long names can be confusing to read.

The nice thing about the `#DEFAULT` section is that the Ctrl-F keystroke loads the Clarion formatter, so I was able to design a window visually. Mine is a bit crappy, but you can design your own, and it will be available whenever you use the template.

Note that `#DEFAULT` structures can be edited from the Template Registry by clicking on the Properties button, then the Default button, but I *don't* recommend this because it will add quite a bit of extraneous code to your template.

Including the API declaration

Including the API code to disable the window is straightforward. All that needs to be done is find the embed I used in the source code example and make sure the template places the code there. In this case I needed two embed points…

First, at the `GlobalMap` embed, which gets added to the Global module in the data section:

```
#AT(%GlobalMap),PRIORITY(4000), WHERE(%GWBIncludeDisableWindowAPI = 1)
 MODULE('GWBFBW_Template')
  GWBDisableWindow(USHORT,SHORT Enabled=False),PASCAL,NAME('EnableWindow')
 END
#ENDAT
```

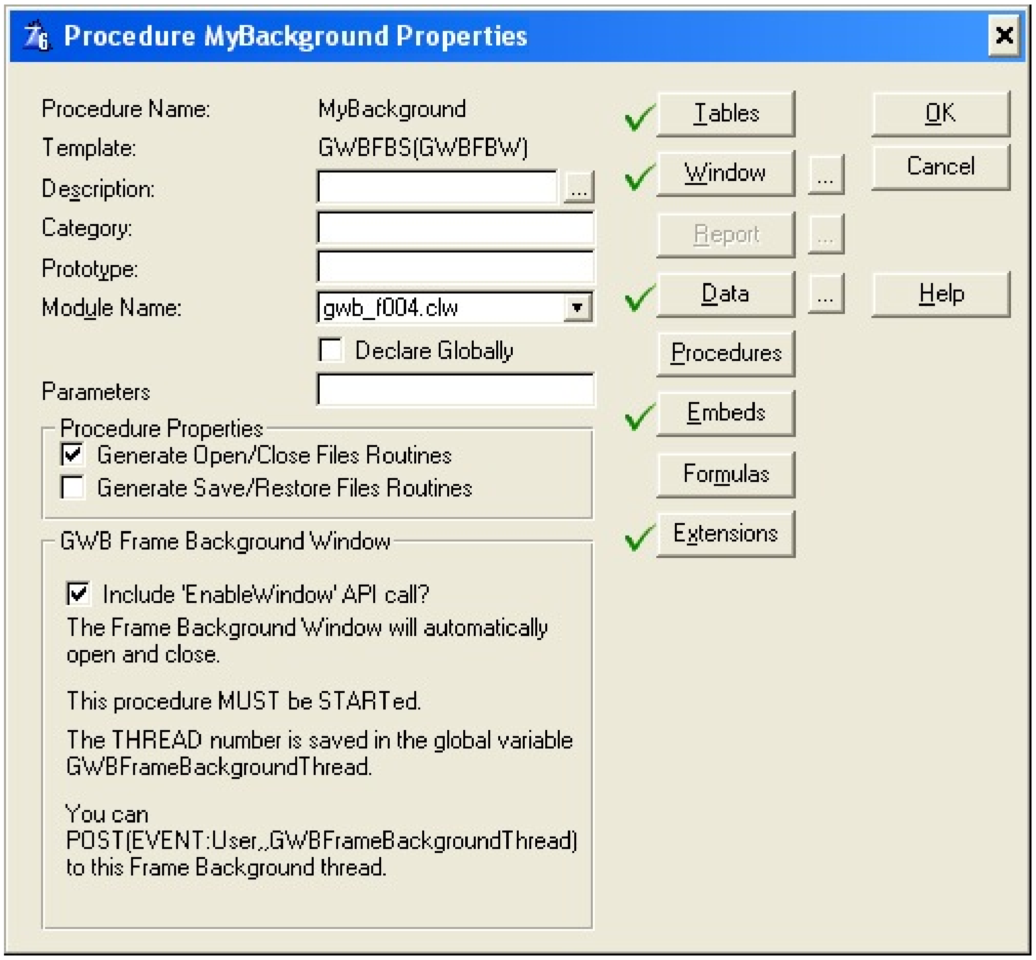
Second, since this is a template for a Source procedure I need to add my code at the Processed Code embed point. The following code checks to make sure the procedure was `START`ed, saves the current `THREAD` number to a global variable declared by the template, `OPEN`s the window, and finally calls the API function to disable the window:

```
#!---------------------------------------
#AT(%ProcessedCode),PRIORITY(4999)
#!---------------------------------------
  IF GlobalErrors.GetProcedureName() <> ''
    MESSAGE('The Procedure ''%Procedure'' should have been ' &|
          'STARTed, it won''t be displayed!','Error',Icon:Exclamation)
    RETURN
  ELSE
    GlobalErrors.SetProcedureName('%Procedure')
  END
  GWBFrameBackgroundThread = THREAD()
  #EMBED(%BeforeOpenWindow,'Before Open Window')
  OPEN(%Window)
  #EMBED(%AfterOpenWindow,'After Open Window')
  GWBDisableWindow(%Window{PROP:Handle})
  #EMBED(%AfterDisableWindow,'After Disable Window')
```

I’ve added several `#EMBED` points so I can add plenty of extra code if the mood takes me. Following this code is my `ACCEPT` loop; check the template to see what embeds I have allowed for there.

#PROMPTs for the template

Near the beginning of most templates you’ll find a series of `#PROMPT` statements. These are used to provide a bit of information and options to the programmer at design time (see Figure 1). In this case I have provided options to include the file opening and closing routines, and include the API declaration itself. The latter is usually provided to prevent the same API being declared from more than one template.



Procedure Name: MyBackground
Template: GWBFBS(GWBFBW)
Description:
Category:
Prototype:
Module Name: gwb_f004.clw
Declare Globally
Parameters

Procedure Properties
[✓] Generate Open/Close Files Routines
[ ] Generate Save/Restore Files Routines

GWB Frame Background Window
[✓] Include 'EnableWindow' API call?
The Frame Background Window will automatically open and close.

This procedure MUST be STARTed.

The THREAD number is saved in the global variable GWBFrameBackgroundThread.

You can POST(EVENT:User,,GWBFrameBackgroundThread) to this Frame Background thread.

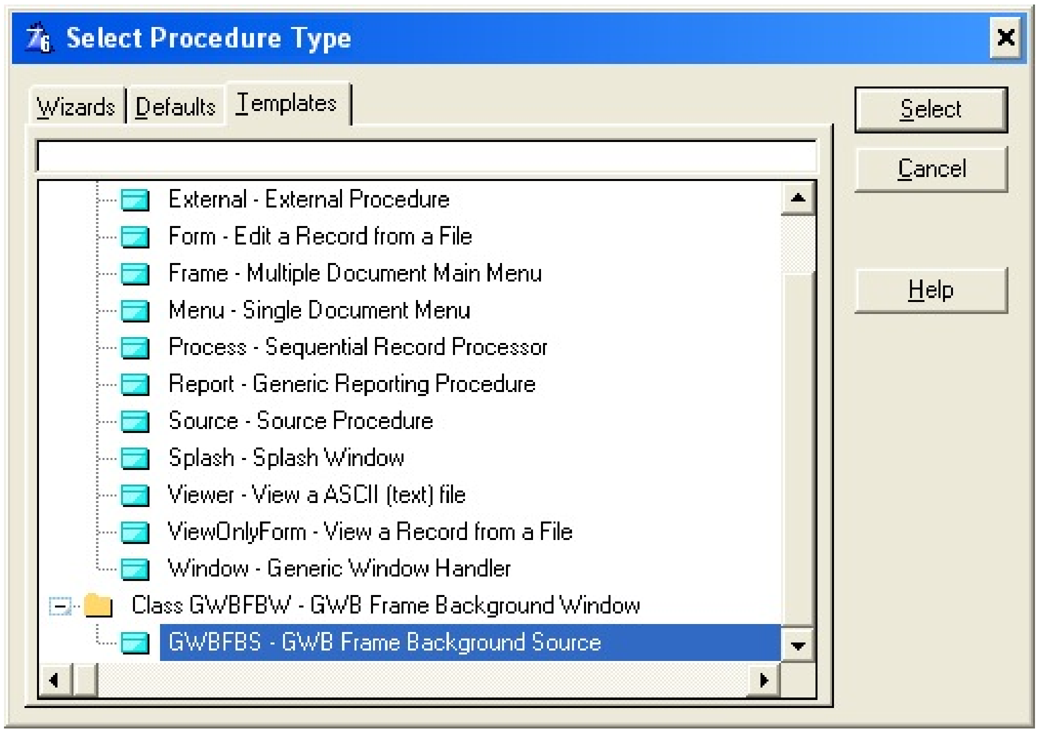
Figure 1. Splash procedure template prompts

`#PROMPT`s are often an area where a template writer can spend an unnecessary amount of time "prettying-up" a template. C7 and C.Net are likely to have a Template Window formatter, so hopefully this complaint will be a thing of the past!

That largely completes the template writing process for this template. So how do you use the template?

Implementing the template

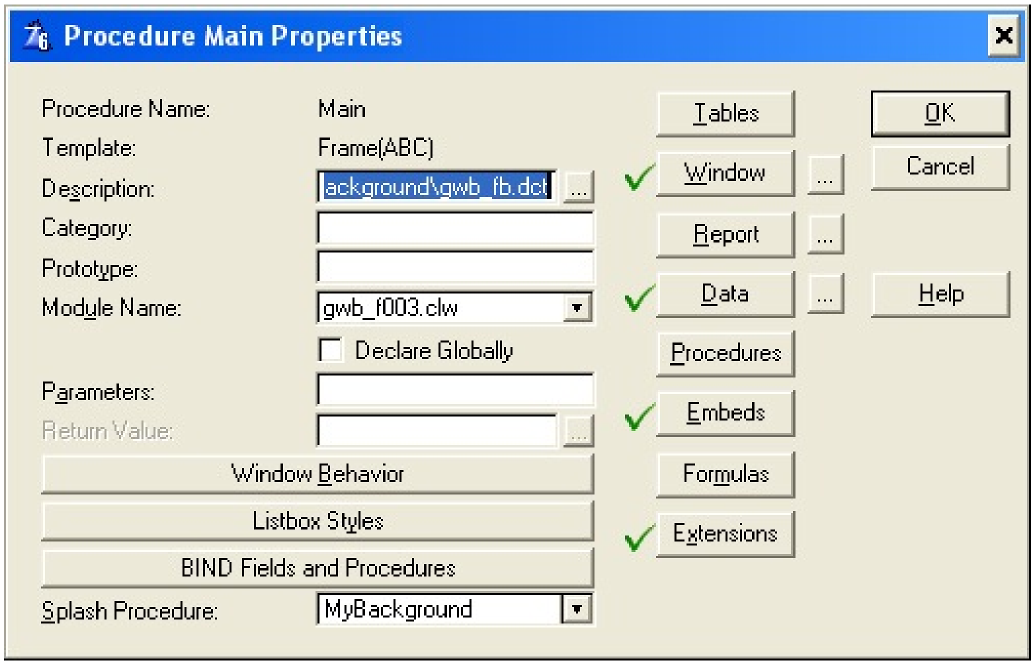
If you haven't done so already, copy GWBFrameBackground.Tpl from the downloadable source zip into your favorite template folder, open the Clarion template registry and register this template. Open an app, press the Insert key to add a new procedure, give it a name such as `MyBackground`, and choose the GWBFBS procedure type, as in Figure 2.



External - External Procedure
Form - Edit a Record from a File
Frame - Multiple Document Main Menu
Menu - Single Document Menu
Process - Sequential Record Processor
Report - Generic Reporting Procedure
Source - Source Procedure
Splash - Splash Window
Viewer - View a ASCII (text) file
ViewOnlyForm - View a Record from a File
Window - Generic Window Handler
Class GWBFBW - GWB Frame Background Window
GWBFBS - GWB Frame Background Source

Figure 2. Choosing the GWBFBS procedure type

From your Frame procedure, specify this new procedure as the Splash procedure for your application (Figure 3).



Procedure Name:	Main
Template:	Frame(ABC)
Description:	ackground\gwb_fb.dct
Category:	
Prototype:	
Module Name:	gwb_f003.clw
Declare Globally	☐
Parameters:	
Return Value:	
Splash Procedure:	MyBackground

Figure 3. Setting the splash procedure

This is the quickest way to implement this template and ensure it is STARTed correctly. An example of the template implemented without any additional source code is found in **gwb_fb4.app.**

Final Example

Just for fun, I played around with the template by adding some source code, enabling and disabling the window, using timer functions, checking what `EVENT`s the window received etc, to see what worked and what would break. The **gwb_fb5.app** is the result, including some interesting anomalies, which I'll let you discover for yourself.

Addendum

Since beginning work on this article another post by Arnór Baldvinsson and/or Jim Kane has mentioned another API technique which changes the frame background color. I haven't experimented with this technique yet, but it looks like it could be a useful future addition to this template. The basic idea is to use the `GetClassLong` and `CreateSolidBrush` API calls to assign a custom background brush to the window during the `ThisWindow.Init` call.

Conclusion

Simply changing the Frame background can dramatically enhance the appearance of, and add value to, an application. This article discusses some simple techniques and explains the process of writing a "10 minute template" (well, under ideal conditions, anyway) to implement one of these techniques quite simply. Writing the template, and this article, has been a valuable learning experience for me; if anyone wishes to improve upon the template I would welcome all suggestions.

Download the source

Reader Comments

Add a comment

- » Geoff, If you can create this template is 10 minutes,...

Clarion Magazine

IsNetworkAlive: When You Don't Need to Know *That* Much

by Steven Parker

Published 2005-12-23

My colleague Henry Plotkin recently examined ways of checking whether a client PC was able to talk to another, named, PC on the network. However, I think he liked the `IsDestinationReachable` API so well that gave short shrift to the `IsNetworkAlive` call.

IsDestinationReachable, the Practical Side

The problem with `IsDestinationReachable` is that it requires specific information about the other computer. I need to know the IP address or the UNC name of the server. That means either that the end user must tell me or I must compute what I need to know for the API call.

My users can be described (charitably) as "PC illiterate and bound and determined to stay that way." They wouldn't know an IP address from a deadly disease on a bet. I don't think any know what a UNC name is.

Compute the remote computer's identity myself? Well, yes, I do have a datum in a configuration file with the path to the data directory. Typically it is composed of a mapped drive letter, a parent directory and the data directory. Real examples look like:

```
    N:\ARBAPRO\DATA\
    F:\EVANSTON\MAIN\ARBAPRO\DATA\
    I:\TECH-GRP\ARBA\ARBAPRO\DATA\
```

Etc.

In a (very) few cases, UNC names are used. This happens when the user has not mapped a drive and uses my "Select Path to Data" utility to create the datum. It also happens when they have a mapped drive and forget to select it from this setup utility.

However, I discourage using UNC names. Some time ago, Paul Attryde posted a message on comp.lang.clarion stating that he had found UNC paths to be less stable, degrading over time, than mapped paths. That was enough for me.

I suppose I could take the data path and parse it myself. If the first character is "C," then the program is running locally. Therefore, the "host" is 127.0.0.1.

If the first two characters are "\\," then I have UNC name. I can use `INSTRING` to find the next backslash and extract the portion of the datum that I actually need. (Remember, as Henry pointed out, final backslashes will cause the API call to fail.)

If the first character of the data path datum is a number, I have an IP address. I "only" need to truncate it to the numeric portion.

But what if the first character is a mapped drive letter? When my configuration variable looks like the examples above, life gets exciting.

How do I extract a UNC name or IP address from "`N:\ARBAPRO\DATA\`"?

Another colleague, Marty Portner, observes that `NET USE` with no parameters will return a connection list. This list may be empty:

```
New connections will not be remembered.


There are no entries in the list.
```

It may contain information on every network resource touched:

```
New connections will not be remembered.

Status Local Remote Network

---------------------------------------------------------------------------

OK Z: \\PRINTSERVE\C Microsoft Windows Network

Disconnected \\KATHIE\C Microsoft Windows Network

OK \\PRINTSERVE\C Microsoft Windows Network

The command completed successfully.
```

Or, it may contain just what I need to know:

```
New connections will not be remembered.

Status Local Remote Network

---------------------------------------------------------------------------

OK P: \\MILTON\Milton-C Microsoft Windows Network

The command completed successfully.
```

Regardless, if there is a map, the mapped drive letter will show up somewhere in the list ("Z" and "P" in the examples above are mapped drives).

Now all I need to do is get the user to run `NET USE` (okay, I *can* Run(‘net use',1) for them; well, no, I can't as it looks like `NET USE` is a built-in and requires the command processor for a `Run`). Then "all" they need to do is enter the information for me.

Right.

All that's left is to output `NET USE` to a file and sequentially read the records in the file looking for the mapped drive letter. Then I can parse out what I need. (Unfortunately, in some fast and dirty experiments, I could not get `Run('cmd.exe net use > net.txt',1)` to work.)

Frankly, though I know that ultimately I'll have to do this, it sure does sound like *work*. (I did notice that the drive letter, if present, is at column 15 and the UNC names begin at 24 should you feel motivated to undertake this task before I do.)

IsNetworkAlive to the Rescue?

Microsoft describes this API as follows:

> The **IsNetworkAlive** function determines whether or not a local system is connected to a network, and identifies the type of network connection, for example, a LAN, WAN, or both.

Because `IsNetWorkAlive` checks only that a network connection, any network connection, can be made and not a connection to a particular file server, it appears to be less useful than `IsDestinationReachable`.

However, MSDN further comments:

> This function can be used by an application to determine whether or not there is network connectivity before proceeding with network operations. A directory service type of application, e-mail client, or Internet browser can adapt to various types of network connectivity. For example, a printing operation can be deferred until a network connection is available.

The first sentence gets my attention. `IsNetworkAlive` can be used at program start up to decide whether or not I even need to check a file server. Further, `IsDestinationReachable` can take several seconds, in my experiments, to return if the destination is not reachable. When I turn off my router, `IsNetworkAlive` returns instantly.

So maybe there is something to look at here.

Further, many of my users have networks that only run my software. So I am often in a situation where there cannot be multiple servers (thus risking a "false positive"). That is, having the server name isn't required for these dedicated networks. Even in larger organizations, where there are multiple servers and my user's computer may use more than one server, the most frequent issues are the local router, NICs and cables. Again, in these kinds of cases, just checking whether any kind of connection is working may be enough.

So maybe there *is* something to look at here.

The Technicalities

Microsoft prototypes `IsNetworkAlive` as:

```
Bool IsNetworkAlive{
 LPDWORD lpdwFlags
} ;
```

The function returns `True` or `False`.

For two reasons, this is one of the most difficult APIs to translate into Clarion that I have found.

First, I got fooled by the `LPDWORD`. This made me think that the Clarion prototype should be:

```
IsNetworkAlive(ULong),Long,pascal
```

This did not work. Careful examination revealed that I had been fooled by the use of all uppercase letters. I am used to seeing `lpDWORD` when Microsoft wants a pointer to an integer.

Unfortunately,

```
IsNetworkAlive(*ULong),Long,pascal
```

didn't work either. I kept getting unknown function errors. Finally, I tried

```
IsNetworkAlive(*ULong),Long,pascal,Name('IsNetworkAlive')
```

Et voilá.

Next, include the sensapi.lib Henry provided with his articles. Now it compiles.

On reflection, I think this is one of the dumbest function declarations I have seen in a while. It both returns `True`/`False` *and* the type of network connection, if any, detected.

MSDN states:

> The type of network connection that is available. This parameter can be one of the following values:
>
> NETWORK_ALIVE_LAN
>
> The computer has one or more LAN cards that are active.
>
> NETWORK_ALIVE_WAN
>
> The computer has one or more active RAS connections.
>
> NETWORK_ALIVE_AOL
>
> This flag is only valid in Windows Me/98/95.
>
> Indicates a computer is connected to the America Online (AOL) network.

In fact, `lpdwFlags` is not a passed parameter. It is an integer which the function, if it succeeds, sets. That is, whatever value is in the variable when passed to the function is irrelevant:

```
If IsNetworkAlive(NetworkType)
  ! Network is active
  Case NetworkType
  Of 1
    LAN = True
  Of 2
    WAN = True
  Of 4
    ! AOL
  Else
    ! unknown
  End
Else
  ! network is barf-o
End
```

instead of the much more logical:

```
If IsNetworkAlive(NetworkType)
  Case NetworkType
  Of 1
    LAN = True
  Of 2
    WAN = True
  Of 4
    ! AOL
  Else
    ! el barf-o
  End
End
```

In Microsoft's version, first I test the return value then I check the network type. In my version, it's done all in a go.

In any case, there is a full implementation in the sample app, downloadable below.

Caveats

Testing both Henry's app and the app for this article, I started to notice inconsistent results.

I tested on my home, wireless network. I could "reach" all the IP addresses my router's DHCP service had assigned. But I could not reach all the UNC names I knew corresponded to those addresses.

I turned off the network card in one computer. I could still "reach" it using `IsDestinationReachable`.

I am hardly a networking expert. In fact, I consider myself only minimally competent in networking. But it looks to me like both `IsDestinationReachable` and `IsNetworkAlive` are querying the router, the DHCP table, and not necessarily the actual remote computer.

Summary

For many networks, especially dedicated and department nets, `IsNetworkAlive` may be all that's needed. Implementing it doesn't require knowing anything about the file server.

The only downside is that `GetLastError` always seems to return zero (no error), at least in my tests. Therefore, if `the network is not alive, FormatMessage`, as Henry described it, won't give any additional information.

Download the source

Steve Parker started his professional life as a Philosopher but now tries to imitate a Clarion developer. He has been attempting to subdue Clarion since 2007 (DOS, that is). He reports that, so far, Clarion is winning. Steve has been writing about Clarion since 1993.

Reader Comments

Add a comment

- » IsNetworkAlive(*ULong),Long,pascal,Name('IsNetworkAlive')
- » You could create a batch file that runs the net use...
- » Good one. Hadn't thought of using a batch file.

Clarion Magazine

Translating Mapped Drives To UNC

by Steven Parker

Published 2005-12-23

I recently complained that I would have to translate mapped drive letters to UNC names so that I could check whether or not network resources were available. Paul Attryde, with his characteristic unstinting help, pointed me to a fairly easy and very solid solution.

Paul's solution is a combination of two API calls. First, `GetDriveType` to determine whether a drive is remote (networked) or not. Next, `WNetGetConnection` which "retrieves the name of the network resource associated with a local device." Parsing the result down to the server name only is quite easy at that point. Then, with the UNC server name, `IsDestinationReachable` is easy to implement in code.

GetDriveType

Microsoft prototypes GetDriveType as

```
Microsoft prototypes GetDriveType as
UINT GetDriveType {
      LPCTSTR lpRootPathName
} ;
```

The Clarion prototype is:

```
GetDriveType(*CString),Long,Pascal,Raw,Name('GetDriveTypeA')
```

Because `GetDriveType` is in Kernel32, no additional lib files need to be included.

`GetDriveType` can return one of seven values:

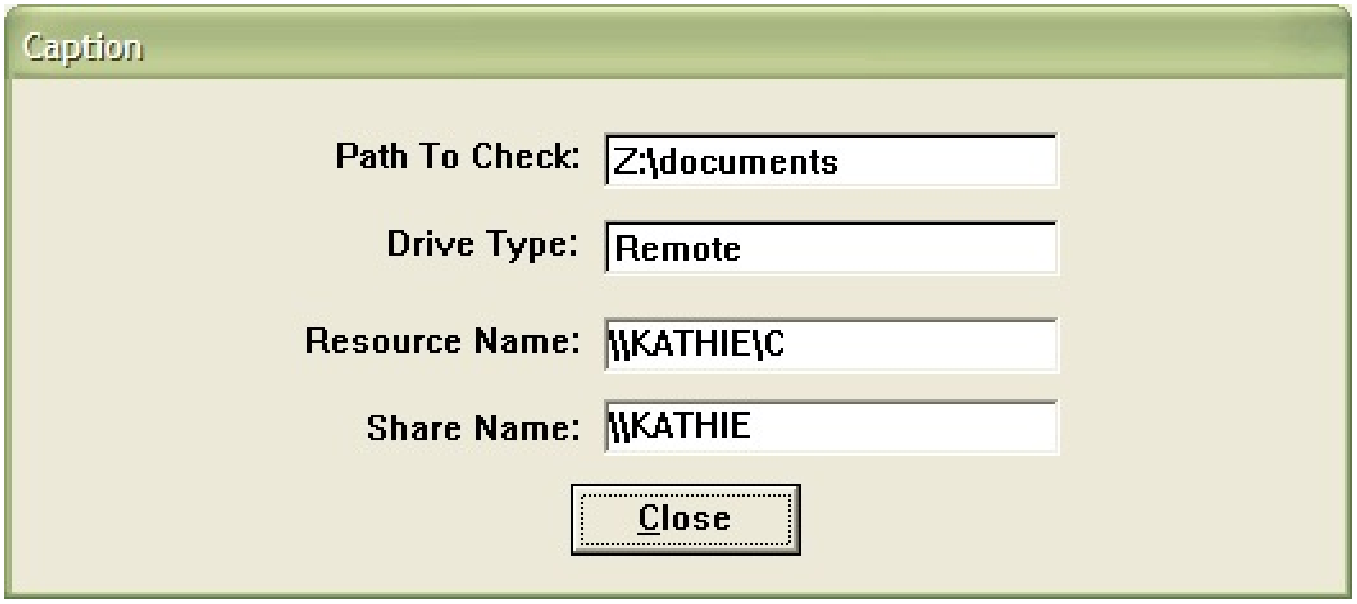
Value	**Equate**
`DRIVE_UNKNOWN`	0
`DRIVE_NO_ROOT_DIR`	1
`DRIVE_REMOVABLE`	2
`DRIVE_FIXED`	3
`DRIVE_REMOTE`	4
`DRIVE_CDROM`	5
`DRIVE_RAMDISK`	6

`DRIVE_NO_ROOT_DIR` in Microsoft-speak is "The root path is invalid, for example, no volume is mounted at the path." In English, this means "not found."

When I ran the included test program and entered a non-existent drive letter, I got `DRIVE_NO_ROOT_DIR`. `GetDriveType` also correctly identified my entry of C:\, my entry of the drive letter for an external USB HDD, my entry of the letter for my CD drive and my entry of the letter for a USB flash drive.

So far, so good. In fact, very good.

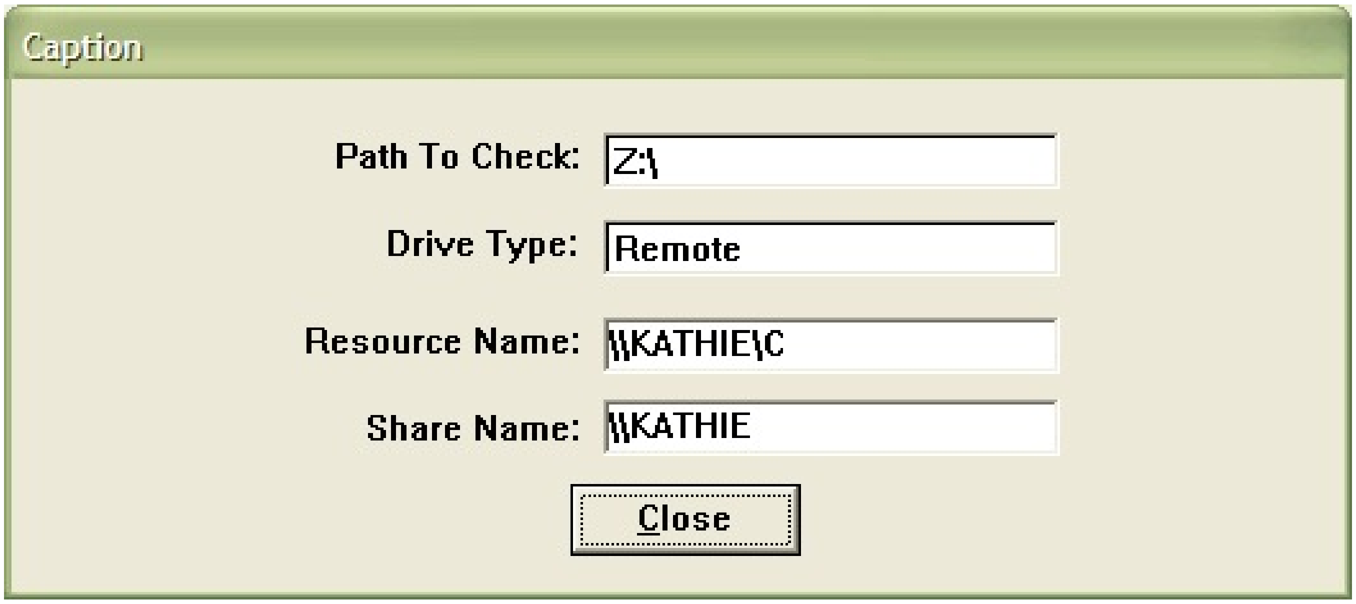
Then, I mapped a drive letter to another PC. When I entered that letter (the letter and the colon are actually required; if a path is entered, the code parses out the first two characters), I got the UNC path. From that, I was able to get the UNC server name:



Path To Check: Z:\documents
Drive Type: Remote
Resource Name: \\KATHIE\C
Share Name: \\KATHIE

Figure 1. GetDriveType at work

Note that the relevant parts of the results are the same with only the drive letter and colon:



Path To Check: Z:\
Drive Type: Remote
Resource Name: \\KATHIE\C
Share Name: \\KATHIE

Figure 2. Minimal data entered

The code that gets the drive type is:

```
DriveToCheck = PathToCheck[1:2]
RetVal = GetDriveType(DriveToCheck)
Case RetVal
Of 0    ! DRIVE_UNKNOWN
  DriveType = 'Unknown'
Of 1    ! DRIVE_NO_ROOT_DIR
  DriveType = 'Path not found'
Of 2    ! DRIVE_REMOVABLE
  DriveType = 'Removable'
Of 3    ! DRIVE_FIXED
  DriveType = 'Fixed drive'
Of 4    ! DRIVE_REMOTE
  DriveType = 'Remote'
Of 5    ! DRIVE_CDROM
  DriveType = 'CD ROM'
Of 6    ! DRIVE_RAMDISK
  DriveType = 'RAM disk'
End
```

(A frequently asked question on the newsgroups is "how do I get the drive's free space?" Check the end of the Microsoft documentation of GetDriveType.)

When GetDriveType returns 4, the drive is networked and I need to use `WNetGetConnection`.

WNetGetConnection

The `WNetGetConnection` API call is used to get "the name of the network resource associated with a local device."

This function takes the name of the local device, the address of a string to receive the resource name and the length of that string.

The Clarion prototype is:

```
WNetGetConnection(*CString lpLocalName,*CString |
lpRemoteName,*ULong),ULong,Raw,Pascal,Name('WNetGetConnectionA')
```

However, `WNetGetConnection` is not in any of the Windows DLLs usually included by Clarion. It is in MPR.DLL. I am including the LIB I made with LibMaker (that was a real adventure; huge numbers of functions appear to be duplicated as far as Clarion is concerned).

So, if `GetDriveType` returns 4, I execute this code:

```
lpnLength = Size(lpRemoteName) ! compute third parameter
GetCon = WNetGetConnection(DriveToCheck,lpRemoteName,lpnLength)
If GetCon = 0            ! no error lpRemoteName contains UNC path
  x# = Instring('\',lpRemoteName,1,3) ! parse out anything
  ShareName = lpRemoteName[1 : x#-1]  ! that is not server name
Else          ! error
  WNetGetLastError(GetCon,lpErrorBuf,Size(lpErrorBuf), |
    lpNameBuf,Size(lpNameBuf))
  Message('Error code: ' & GetCon & '<13,10>' & |
    lpErrorBuf,'Error',ICON:Hand)
End
```

`ShareName` can then be passed to `IsDestinationReachable` to complete the test.

Summary

So, there you have it. With the help of someone who knows what he’s doing, going from a drive letter to a UNC name really isn’t all that difficult.

Download the source

Steve Parker started his professional life as a Philosopher but now tries to imitate a Clarion developer. He has been attempting to subdue Clarion since 2007 (DOS, that is). He reports that, so far, Clarion is winning. Steve has been writing about Clarion since 1993.

Reader Comments

Add a comment

Clarion Magazine

Clarion Apps Can Be Sexy!

by Colin Wynn

Published 2005-12-30

In today's market I believe that applications don't just need to work; they need to be user friendly, with an attractive, standardized user interface.

Now before I continue I'd just like to say the views expressed in this article are my own personal ones, so if I sound like I'm knocking your application then I apologise now. But do take note of this article; it doesn't take much to spruce up your application(s), you just need to set a few ground rules, make a standard, and *strictly* adhere to it!!!

From looking at current applications on the market today, talking to fellow developers, and taking notes from Windows User Experience: Official Guidelines for User Interface Developers and Designers (ISBN 0735605661) by Microsoft Press, I created what I call my *Development Bible*. It's a 15 page document that I refer to on a regular basis. In it I list how toolbars, browses, forms, wizards and reports should look, right down to the size, position, look and feel of controls, and I'm going to share *some* of my Bible with you.

As time has gone on my Bible has gotten smaller as I've templated a lot of my standards, so when I design a browse, form, wizard or report I just drop in my templates that control everything that's common.

So let's look at a few of my standards, starting with some global ones:

- Windows should always use the font Tahoma, Regular style and size 8.
- Reports always use Verdana, Regular style and size 8.
- Apply an icon to every window.
- If the window consists of only one tab, remove the tab and place the

controls within a group, except for action buttons such as Save and Close.
- Tab headings are Book Style Capitalisation, but have *no* hotkey.
- Controls start 4 dialog units from the top of the window.
- Controls start and end 7 dialog units from the left and right edges of a window, except for sheets, which start and end 4 dialog units from the edge.
- Controls end 7 dialog units from the bottom of the window.

The application frame

An application frame (see Figure 1) should use a Resizable frame type and an initial size of Maximised.

Menu items use Book Style Capitalisation, should *always* have a Hotkey assigned; if further input is required, then follow the item with an ellipsis (…). Also detail the message of the action using Sentence style capitalisation.

My Application Frames always have File, Edit, Actions, Reports, Tools, Window & Help submenus.

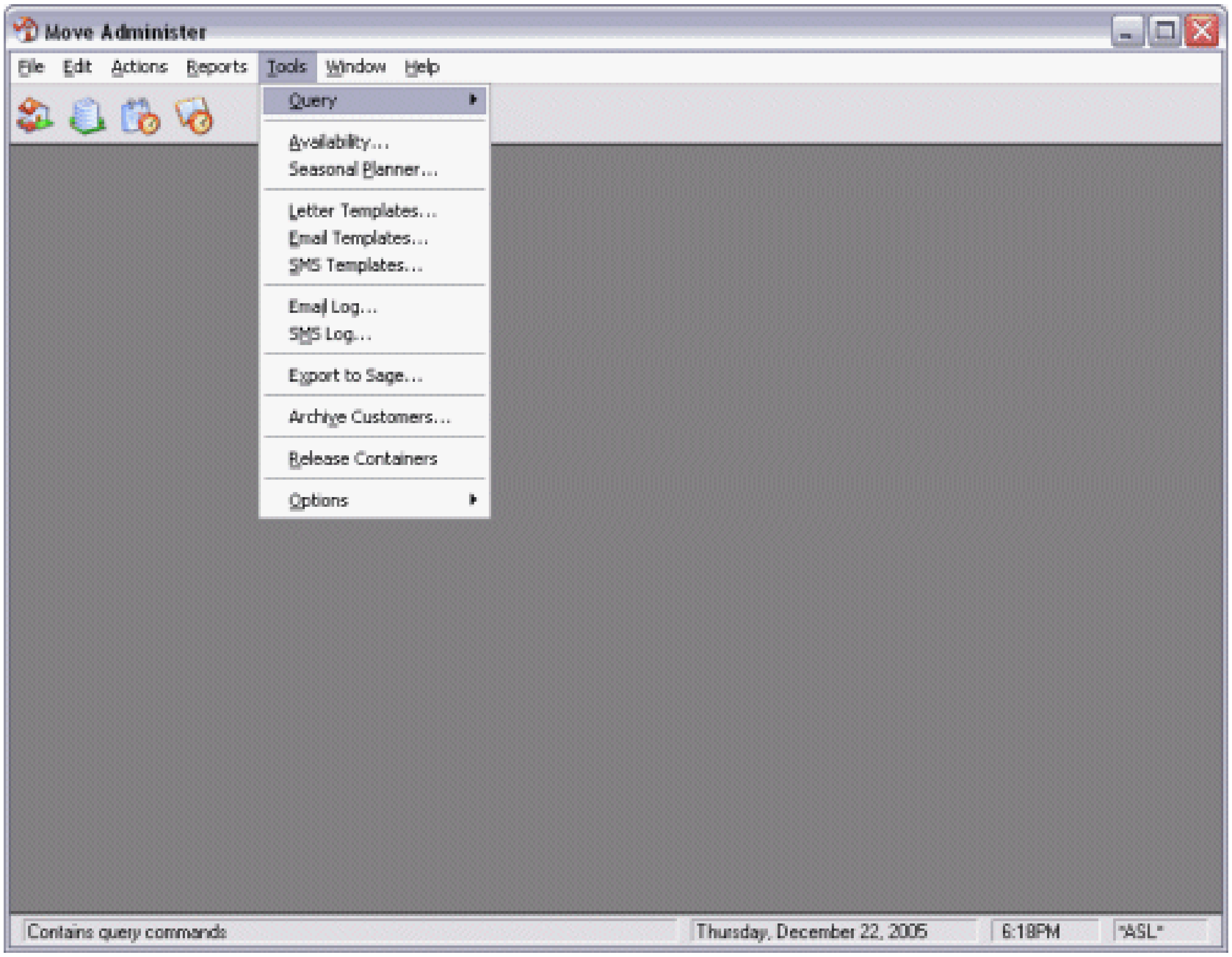


Figure 1. Application frame standards (see full size image)

An application frame should always have a Toolbar (see Figure 2), where you place your most popular menu items in the form of Buttoned Icons, placed in groups (not the control Group) and separated by Dividers (panels).

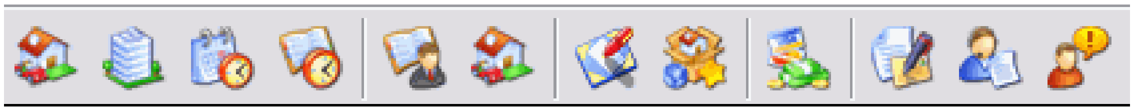
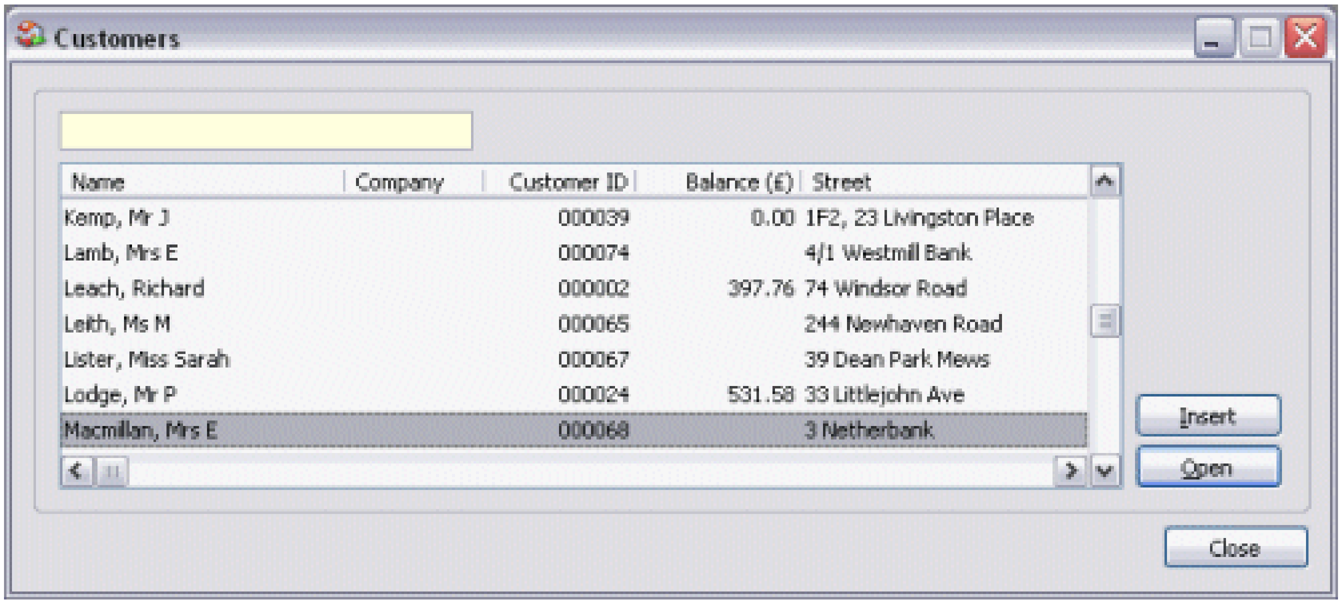


Figure 2. The application toolbar

A toolbar should be 23 dialog units High, and begin with a Divider. Toolbar buttons are *always* Flat, contain *no* text, and should have a Ypos of 2, a Width of 22 and a Height of 21 dialog units, with no spacing between buttons, only Dividers.

The browse

Browses are especially easy to design if you have a standard, as the only items that change are the width of the listbox and window, the xpos of the buttons, and of course the browse column format. With Xplore (a third party add-on by IceTips that I use on all of my browses) the end user can change the browse look and feel. The beauty of my standard is that 90% of the time my browses are identically designed, which means designing a browse is very quick indeed. It's the third party add-ons that I use that allow the end user to change their look and feel.



Name	Company	Customer ID	Balance (£)	Street
Kemp, Mr J		000039	0.00	1F2, 23 Livingston Place
Lamb, Mrs E		000074		4/1 Westmill Bank
Leach, Richard		000002	397.76	74 Windsor Road
Leith, Ms M		000065		244 Newhaven Road
Lister, Miss Sarah		000067		39 Dean Park Mews
Lodge, Mr P		000024	531.58	33 Littlejohn Ave
Macmillan, Mrs E		000068		3 Netherbank

Figure 3. Browse standards (see full size image)

Always allow the use of popup menus, do *not* allow Maximise (only because Clarion doesn't handle this quite right as yet), and have a frame type of Resizable. Rename the Change button to Open (not every user wants to change a record, they want to open it where they can view the contents and then make any changes they require, saving those changes by pressing the Save button or cancelling any changes made by pressing the Close button). Always have a Locator (which is incremental for everything except date and time fields), and place all controls inside a Group except the Close button.

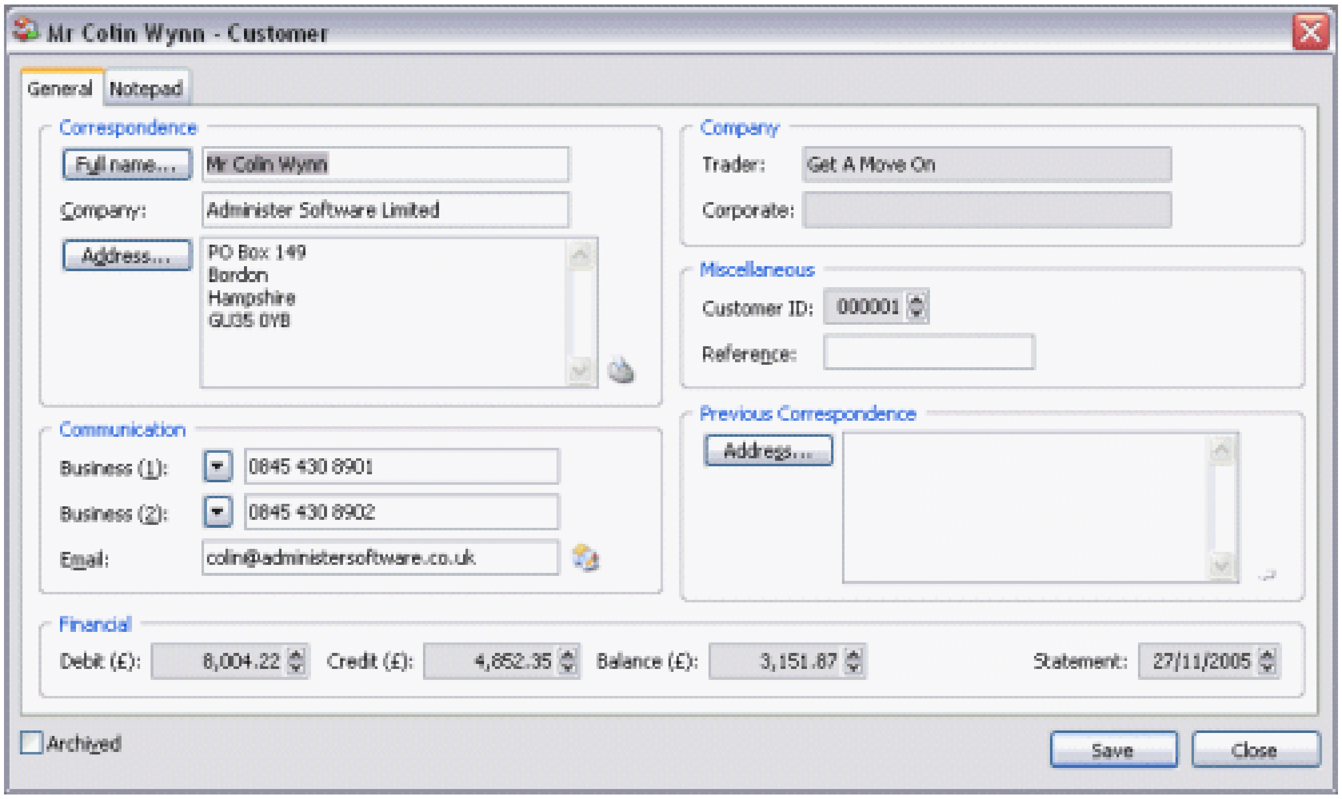
Action buttons, including Insert, Open, Delete, and Close are always 50 dialog units Wide and 14 High. Icons on action buttons are tacky and trashy, *so don't do it!*

I also never have a browse wider that 360 dialog units, and unless the browse has a horizontal scrollbar then the Height is always 92, otherwise is 102.

I'm sure you can now start to see why I can put together a browse quickly, it's only really the width of my browses that change; basically, I know the positions of all the controls off by heart.

The form

Your forms should always offer to save changes, and have a frame type of Double. Set the action buttons to Save and Close, and remember if you only have one tab then remove it, it’s taking up valuable space! If my form has more than three tabs I actually turn the form into a wizard, again using my own templates.



Mr Colin Wynn - Customer

General | Notepad

Correspondence

Full name... Mr Colin Wynn

Company: Administer Software Limited

Address... PO Box 149
Bordon
Hampshire
GU35 0YB

Communication

Business (1): 0845 430 8901

Business (2): 0845 430 8902

Email: colin@administersoftware.co.uk

Company

Trader: Get A Move On

Corporate:

Miscellaneous

Customer ID: 000001

Reference:

Previous Correspondence

Address...

Financial

Debit (£): 8,004.22 Credit (£): 4,852.35 Balance (£): 3,151.87 Statement: 27/11/2005

Archived

Save Close

Figure 4. Form standards (see full size image)

Group all your controls into logical sections.

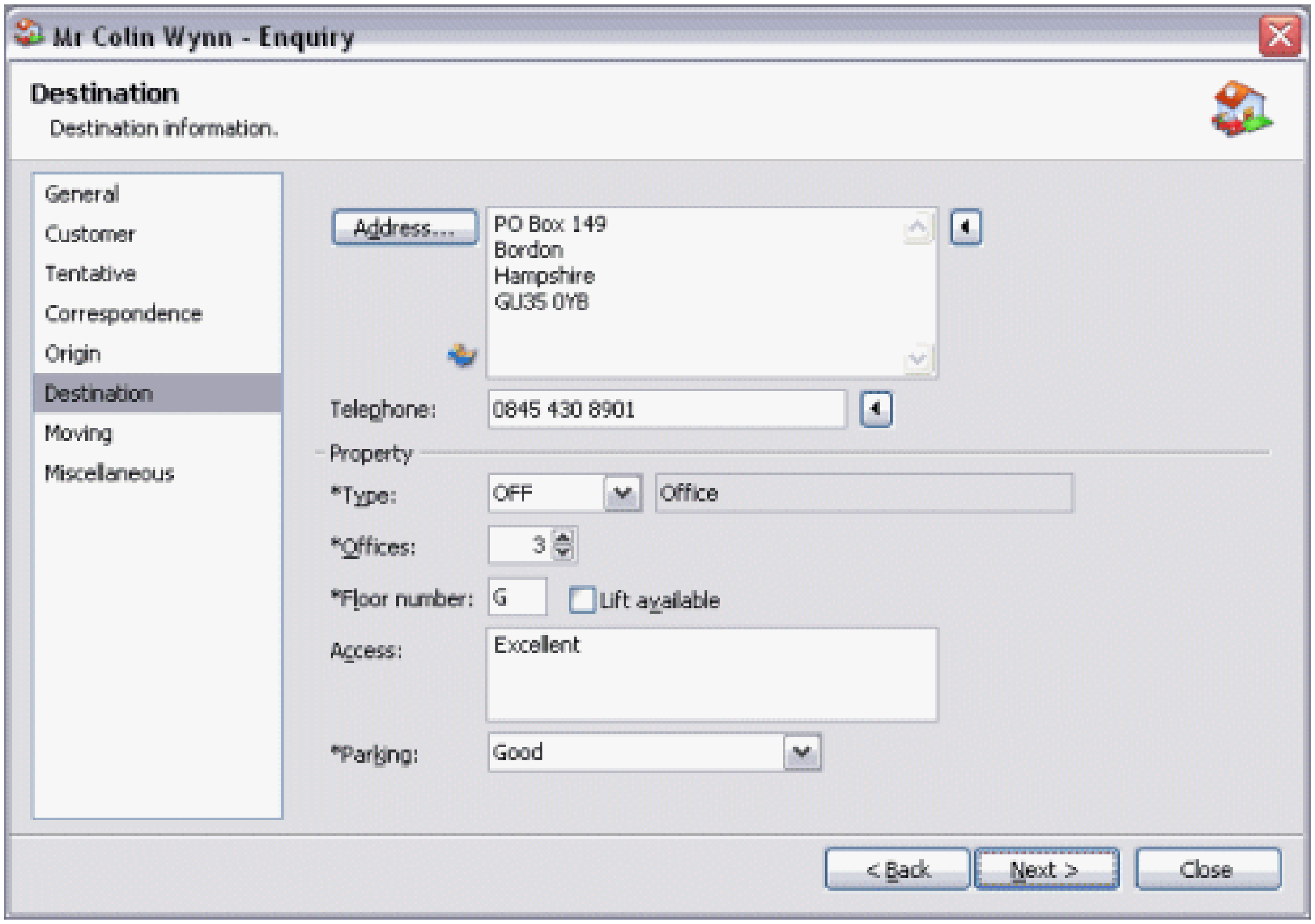
With prompts use Sentence style capitalisation, always end with a Colon, and keep them left aligned. Always assign a Hotkey (active fields only) and make the prompts 8 dialog units high.

Make fields 12 dialog units high and avoid setting colours. To highlight something use `COLOR:HIGHLIGHT`, and when a field is read-only use `COLOR:BTNFACE`. And when in a read-only state always have Skip set and *never* have a hotkey assigned to the field’s prompt.

With buttons use Book Style Capitalisation; if more input is required if pressed, then *always* follow text with an ellipsis (…).

Wizards

My wizards always have a wizard header that sits at the top of the window and consists of a Box, Image, Shadow (Prompt), Title (Prompt), Description (Prompt) and a Panel control.



Mr Colin Wynn - Enquiry

Destination

Destination information.

General
Customer
Tentative
Correspondence
Origin
Destination
Moving
Miscellaneous

Address...	PO Box 149 Bordon Hampshire GU35 0YB
Telephone:	0845 430 8901
Property	
*Type:	OFF Office
*Offices:	3
*Floor number:	G ☐ Lift available
Access:	Excellent
*Parking:	Good

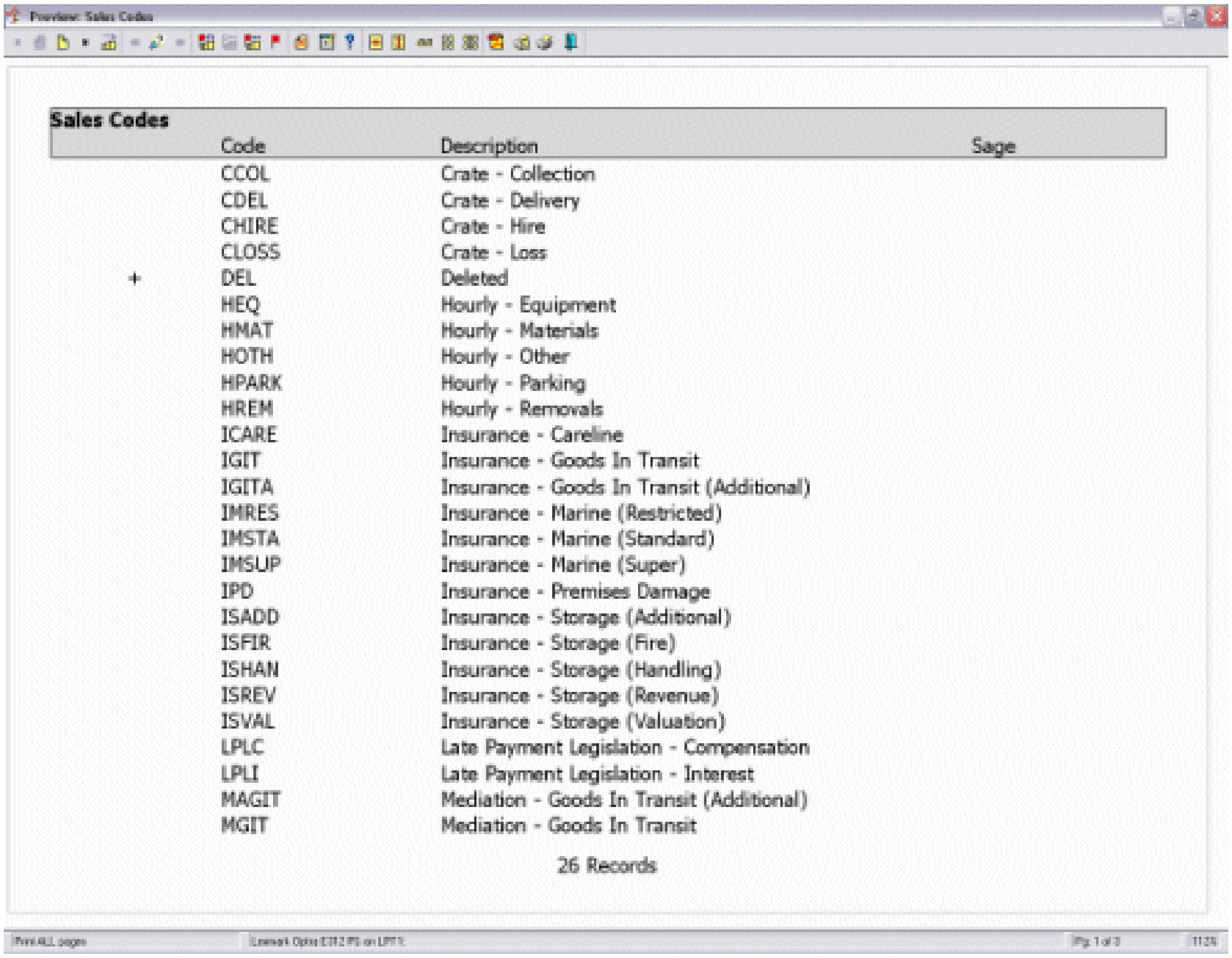
< Back | Next > | Close

Figure 5. Wizard standards (see full size image)

Wizards should always have Next and Back buttons for ease of navigation, although my wizards also use a browse on the left of the window that acts like tabs (its actually a queue generated from the actual tabs created on the window that are then set to wizard so that they don't show). And remember, when using a wizard for Inserting the Finish button should never appear until the last tab, but when used for Opening the Finish button should always be available.

Reports

I don’t have many report standards, although I have default layouts that I’ve added to DEFAULTS.CLW to set to the width and height that I require. So when producing reports I let the wizard create the procedure, and then I delete the window and report and replace with my own from the above mentioned file. But having said that, I do constantly design reports in 1/1000 of an inch (as it’s much more accurate than mm), and I ensure that every report has page numbers and the date of creation.



Sales Codes

Code	Description	Sage
CCOL	Crate - Collection	
CDEL	Crate - Delivery	
CHIRE	Crate - Hire	
CLOSS	Crate - Loss	
DEL	Deleted	
HEQ	Hourly - Equipment	
HMAT	Hourly - Materials	
HOTH	Hourly - Other	
HPARK	Hourly - Parking	
HREM	Hourly - Removals	
ICARE	Insurance - Careline	
IGIT	Insurance - Goods In Transit	
IGITA	Insurance - Goods In Transit (Additional)	
IMRES	Insurance - Marine (Restricted)	
IMSTA	Insurance - Marine (Standard)	
IMSUP	Insurance - Marine (Super)	
IPD	Insurance - Premises Damage	
ISADD	Insurance - Storage (Additional)	
ISFIR	Insurance - Storage (Fire)	
ISHAN	Insurance - Storage (Handling)	
ISREV	Insurance - Storage (Revenue)	
ISVAL	Insurance - Storage (Valuation)	
LPLC	Late Payment Legislation - Compensation	
LPLI	Late Payment Legislation - Interest	
MAGIT	Mediation - Goods In Transit (Additional)	
MGIT	Mediation - Goods In Transit	

26 Records

Figure 6. Report standards (see full size image)

Summary

All in all, with my Bible my applications are completely standard throughout, and although I do adhere to strict guidelines set out by Microsoft I still believe that in places

I've improved upon them. All the applications I design have a common user interface, so designing windows, whether browses or forms, is an absolute breeze. And regardless of whether I develop the application or a member of my team is doing the work, I'm safe in the knowledge that all windows adhere to the standards in my Bible.

Links

- Windows User Experience can be found on MSDN
- All of the icons in my applications have been designed by Jono Hunt of Fooods Design Services.
- The XP look and feel of all my applications would not be possible without the excellent Power-XPTheme product by Poweroffice .
- I don't design a single browse without using Xplore from IceTips. Xplore provides many features that allow my end users to customise every browse to their own individual styling, as well as custom column sorting, filtering and producing many means of outputting data in the format of graphs, reports, CSV and HTML (which actually saves me from producing many report procedures).
- Every report I design is done so with RPM (Report Presentation Manager) from DeveloperPLUS, and I can't wait to get my hands on the forthcoming release with its vastly updated look and feel.

Colin Wynn has been using Clarion since 1990 and runs his own software house in the UK. A former TeamTopspeed member, Colin used to run the largest Clarion-related bulletin board in Europe, and he was the president of the UK Clarion User Group in the early 90s.

Reader Comments

Add a comment

- » Nice lay-out it's something I love. But No resize ? "only...
- » The browses and the application frame are all resizable,...
- » Try the Icetips Previwer for a fully customized look and...

- » SV could perhaps integrate these rules in the templates...
- » We have developed a similar 'look and feel' based on the...
- » We have developed a similar 'look and feel' based on the...
- » Nice, but I would recommend you to also consider...
- » I do like the look of that template, but I still cant see...
- » It would be nice to add this to SV's templates, but it...
- » Sorry Arnor, the RPM preview looks fine to me!
- » RPM looks a tad dated, but the *new* super duper one looks...

Clarion Magazine

The ClarionMag Blog

Get automatic notification of new items! RSS feeds are available for:

XML All blog entries
XML All new items, including blogs

Blog Categories

- »All Blog Entries
- »Clarion 7 Clarion.NET
- »Future Articles
- »News flashes
- »Nifty Stuff

Abe Vigoda, still kickin' in FireFox 1.5

Direct link

Posted Thursday, December 01, 2005 by Dave Harms

Yesterday, while browsing Google News, I noticed FireFox 1.5 had been released, something FireFox itself neglected to tell me. Using my favorite FireFox feature, I opened one of the links in a new tab, and...FireFox crashed. I started FireFox again, clicked on the little FireFox home page icon (I'm hoping someone, somewhere can push the envelope and make that icon just a bit more obscure), and got a page that showed absolutely nothing about 1.5. So I manually changed http://www.mozilla.com/firefox/central/ to http://www.mozilla.com/firefox, which is where I found the download. The install went smoothly, something I worried about just a little because I have all of my FireFox configuration settings in a non-standard directory, to make it easier (in my mind, anyway) to transfer my favorites etc to/from my laptop.

I really can't see much difference between 1.0.7 and 1.5, except that so far I haven't encountered any crashes (I guess I can call this FireFix 1.5), and a couple of my extensions stopped working. Happily, Abe Vigoda Status made it through.

You probably remember Abe Vigoda as Detective Sgt. Fish on the show Barney Miller. As a kid I was convinced Barney Miller was a Canadian show because of the dry humor and the lousy production values. At the time Abe looked, to me at least, to be in his 70s, but was in fact about 20 years younger. I guess he looked old to everyone else too, because in 1982 People magazine announced Vigoda had gone to the big squad room in the sky. As with that other funny guy, Mark Twain, the reports of Vigoda's death were greatly exaggerated, and Abe even posed in a coffin reading a copy of that issue of People.

Abe Vigoda Status is a little FireFox extension by Bob Vesterman at Vesterman.com ("Serving all of your Vesterman needs since 1999"). The extension periodically polls some server out there in the cloud for Abe's situation, and displays this in the status bar. Earlier versions just showed the word "alive"; the current version also displays a tiny, but nonetheless flattering, picture of Abe. You can click on the status bar for more options, including Abe's Wikipedia entry.

I realize a few people think this extension is in bad taste. I can only imagine that Abe Vigoda thinks it's a hoot. And I hope to see that word "alive" on my status bar for many years to come.

Crossposted at Knobblegrud

Blog: The Movie

Direct link

Posted Monday, December 05, 2005 by Dave Harms

Okay, even though you're reading this blog, have you ever wondered exactly what a blog is, anyway? TheWebLogProject is an open source, grassroots movie about blogs. Well it isn't a movie yet. It's a collection of video clips, which will be made into a movie.

Unfortunately for me, the clips are all QuickTime, and QT's obviously hosed on my computer, because Robert Scoble's clip crashed both IE and FireFox 1.5. So I went over to Breathless Bob's Blog just for fun. Can't say I learned much else from this experience except that you don't have to be able to write to be a famous blogger.

Crossposted at Knobblegrud.

Help update Clarion's Wikipedia entry

Direct link

Posted Tuesday, December 06, 2005 by Dave Harms

A few days ago Carl Barnes posted a major update to the Clarion Wikipedia entry, which was subsequently updated by a few others, including me. Anyone can edit a Wiki, so go have a look, and if you see something that needs updating, have at it.

For the most part I'm a fan of Wikipedia, and I use it more frequently as time goes on. But as Andrew Guidroz pointed out to me, there is a dark side. John Seigenthaler, a retired journalist and former editorial page editor at USA Today, has written about discovering his Wikipedia biography contained erroneous, and malicious information. He characterized the entry by an unknown person as "Internet character assassination".

You may be led astray by a Wikipedia entry, since anyone can make changes. But at the least you can see what changes have been made to any entry via the History tab. Here's a comparison of one of my revisions, followed by a correction I made when Viggo Kleven pointed out that Clarion 1.1 did not have a dongle.

Clarion.NET, C7 alphas begin in January

Direct link

Posted Wednesday, December 07, 2005 by Dave Harms

Replying in a thread about the Planet Clarion podcasts, Bob Z stated yesterday that alpha testing for Clarion.NET *and* C7 will begin with a select group in January. Z also said that Clarion.NET "can be used" with WebForms *and* WinForms.

Whether WebForms will be there for the alpha isn't clear to me, but it's great news that it's in the works.

Look for a Planet Clarion podcast with Bob Z in the near future.

Multiple monitors and KVMs

Direct link

Posted Thursday, December 08, 2005 by Dave Harms

I've been using dual monitors for some time now (one LCD and once CRT, with a Matrox P650 video card) and at some point I'll probably add another monitor. It's pretty hard to imagine going back to just one monitor, and judging by this survey I took in August, I'm not alone. I use the LCD for whatever I'm currently working on, and I put stuff I just want to keep an eye on (like Skype and the Windows taskbar) over on the CRT. I stretch a few apps (Eclipse and FrameMaker) across both screens.

Brahn Partridge sent me a few interesting links. MaxiVista turns an extra computer (say, a laptop) into an additional monitor. This product (or one like it) has been discussed on the SoftVelocity newsgroups in the past.

Brahn also mentioned Synergy 2, which is a sort of virtual KVM (keyboard, video, mouse) switch that is available for multiple operating systems. Back when I ran the ClarionMag server in-house, I first had a separate monitor and keyboard for every server. Eventually I graduated to a KVM, which gave me some disk space, but which required some expensive and clunky cabling. I eventually gave the KVM to our local Computers for Schools program. What I really needed back then was something like Synergy, which is open source remote control software that runs over TCP/IP. From the Synergy 2 SourceForge home page:

> Synergy lets you easily share a single mouse and keyboard between multiple computers with different operating systems, each with its own display, without special hardware. It's intended for users with multiple computers on their desk since each system uses its own monitor(s).
>
> Redirecting the mouse and keyboard is as simple as moving the mouse off the edge of your screen. Synergy also merges the clipboards of all the systems into one, allowing cut-and-paste between systems. Furthermore, it synchronizes screen savers so they all start and stop together and, if screen locking is enabled, only one screen requires a password to unlock them all.

Supported operating systems include Windows, from W95 to XP, Unix X Windows (includes Linux, Solaris, etc), and even Mac OS X (although the Mac version is not feature complete).

Clarion.NET and exception handling

Direct link

Posted Thursday, December 08, 2005 by Dave Harms

There's an interesting discussion going on over at the Clarion Community web board on exception handling in Clarion.NET. Exception handling is one of the bigger changes in Clarion.NET; in a nutshell, you mark a block of code with a TRY directive, and if a THROW occurs anywhere in that block, execution immediately jumps to the end of the block where the exception can be handled, or thrown further up the calling chain. I gave an example of how I thought exception handling might be written in Clarion (there was an example at DevCon 2004, but I can't recall the specifics). I remarked on how a method that can throw an exception usually has to be marked as such, and Carl Barnes corrected me, pointing out that this is not the case in .NET. Carl also pointed out that exceptions are slow and expensive, and according to the Jan 2006 MSDN mag you shouldn't use them to determine program logic. Carl argues that exceptions can easily become a mine field, and while unhandled exceptions can certainly bubble up to cause problems where least expected, they're still a powerful and, in my opinion, welcome addition to the Clarion toolbox.

Clarion web dev for the masses (of Clarion developers)?

Direct link

Posted Friday, December 09, 2005 by Dave Harms

I meant to post earlier this week on Bob Zaunere's comment in the softvelocity.public.clarion6 newsgroup that when Clarion.NET goes gold, it will be able to create both WinForms and WebForms applications. You probably wouldn't want to do this with a single APP, however. In another message, Z says:

> The UI related code has to change from a desktop to web app. And from a practical point of view you may be better off maintaining 2 apps that share a common Dictionary. But all of the business logic, data validation, user embed code, etc. can be shared by the two types of applications.

The inclusion of WebForms in the final release is excellent news, and brings the promise of a widely used Clarion web development platform. Although we've had numerous incarnations of web development tools from Topspeed/SoftVelocity, none have ever gained wide acceptance in the development community. Of course, until recently the majority of Clarion developers didn't have a burning need for web dev tools. But that's clearly changing.

Now I just want to know is how easy it will be to incorporate AJAX into Clarion WebForms...

The SV blog lives, and so does Clarion/PHP 2.0

Direct link

Posted Monday, December 12, 2005 by Dave Harms

A new post on the SV blog site announces the 2.0 release of Clarion/PHP. I still think that this is a sleeper product, given Clarion's template technology and PHP's massive popularity. New in 2.0:

- drill-down data sets, which really means progressive browse filtering
- embeddable code in the JavaScript functions, as well as custome JS functions; better client-side validation
- file uploads
- integrated email via the PHPMailer class
- ability to specify the ADOdb library location
- inactivity timeout setting
- a bunch of productivity enhancements.

Clarion/ASP will get the same enhancments.

TPS ODBC 5.0 and C6.3

Direct link

Posted Monday, December 12, 2005 by Dave Harms

Also announced on the SV blog site: version 5.0 of the TPS ODBC driver, and the release of Clarion 6.3 to third party vendors this week. According to Z, "Clarion 6.3 delivers some brand-new functionality and some features long asked for."

Eager as I am to see C7/Clarion.NET, I'm never that excited about updates to C6. In fact, I've sometimes thought this must be a drain on C7 development (despite Z's assurances of one team's efforts not impacting another team's schedule). On the other hand, I expect there's a high level of portability between C7 and C6, so it probably doesn't cost much to add some of the new C7 stuff back into C6, and given the major UI improvements, it's also unlikely that putting new features in C6 now will cannibalize upgrades.

Server-side autoinc in Clarion 6.3

Direct link

Posted Thursday, December 15, 2005 by Dave Harms

Bob Z has blogged about the new server-side autoincrementing feature in Clarion 6.3. The problem this feature addresses is this: if you add a record on a server that uses its own autoincrementing, the record buffer in your Clarion app won't have the autoinc field's current value. There are a couple of templates you can find to get around this problem, but until now there's been no standard solution.

What is particularly cool about the 6.3 feature is its flexibility. The MS-SQL, SQLAnywhere and Pervasive.SQL drivers know exactly how to get the autoinc value already; for other drives, you can specify a custom SELECT statement in the driver string. For example, in MySQL this would probably look like `/AUTOINC=SELECT last_insert_id()`. As Bob notes, that lets you use server side autoinc with *any* back end.

December articles

Direct link

Posted Thursday, December 15, 2005 by Dave Harms

Part 2 of Geoff Bomford's frame background article will be up tomorrow. Although this is mainly a template programming article, there's also a lot of gold there for anyone wanting to know more about the Windows API, specifically how to use API calls to color the application frame background.

Steve Parker has written a couple of articles following up on Henry Plotkin's two-parter on detecting the network, and keep an eye out for Colin Wynn's advice on making your Clarion apps look sharp.

I'm still hoping to do a preview of CapeSoft's new Profiler before Christmas, but I'm not going to guarantee it. We're already quite busy around the house with holiday preparations, and this year we're fortunate to have quite a lot of family coming for Christmas; starting next week I'll be out of the office a fair bit. Bruce Johnson warns that Profiler won't be in beta long, after which the price goes from $299 to $399. No, Profiler isn't cheap, but this is one product that has enormous potential to save you time and money.

SoftVelocity backs a winner

Direct link

Posted Thursday, December 15, 2005 by Dave Harms

Andrew Guidroz pointed out this Business Week article on the possible decline of Java, and the growing acceptance of .NET (and the rise of scripting languages). I wouldn't order the headstone just yet, because Java is still strong in corporate environments, and has some important cross-platform advantages. But just a few years ago Java looked like a much stronger contender than .NET, and it would have been easy enough for SoftVelocity, which at the time had (and may still have) a fair bit of Java expertise in house, to pursue Clarion for Java. Instead, Bob Z steered the ship into .NET waters.

That choice is looking better every day.

VS, Clarion.NET, and feeling nostalgic about the "old" Clarion IDE

Direct link

Posted Wednesday, December 21, 2005 by Dave Harms

I've just spent a few days immersed in C# and Visual Studio Express. No, I'm not abandoning Clarion - this was just a small project for another developer, involving a .NET wrapper around an ActiveX component (and part of a larger system involving Clarion and other technologies). If Clarion.NET had been ready, perhaps it would have been the logical choice. But even when Clarion.NET goes gold, I expect many Clarion developers will find a working knowledge of C#, and a free copy of Visual Studio Express, very useful. For one thing, C# is the reference language for .NET, and the vast majority of .NET code you'll see will be in C#, Visual Basic, or both. For another, when you get that sample source, if you can run it in Visual Studio you know you have working code as a baseline for your conversion. Take a moment to answer the Visual Studio survey question on the ClarionMag home page.

Visual Studio is the second "modern" IDE I've used - the other is Eclipse, which I use when I'm writing server-side Java code. So far, I find Eclipse a bit more sophisticated, but that may be just because I'm more accustomed to Eclipse's keyboard shortcuts. And the new Clarion IDE, based as it is on SharpDevelop, will have a similar look and feel to both Visual Studio and Eclipse.

I'm really looking forward to C7/Clarion.NET, so I was a bit surprised when, on running C6 today, I found myself looking at the Windows '95 era design sensibilities of the IDE with more than a little nostalgia.

SoftVelocity bloggers busy

Direct link

Posted Wednesday, December 21, 2005 by Dave Harms

There's been A flurry of activity over at the SoftVelocity blog site. Bob Z has posted two more items about Clarion 6.3 features, one on 6.3's support for SQL Server 2005's Multiple Active ResultSets (MARS), and another on driver-related features. Diego Borojovich weighs in with his first blog entry, on why some people are finding a performance hit with the 2.2 version of the IP driver, and how to fix the problem. And Bob Foreman is back with a preview of the 6.3 help on how Clarion functions used in filters map to back-end SQL code.

Leroy wants your digital photos (updated)

Direct link

Posted Saturday, December 24, 2005 by Dave Harms

Updated with a clarification on how the whole thing came about...

One of the benefits of subscribing to Clarion Magazine is access to the newsgroups, and the busiest by far is Andrews.Kitchen. You won't see much Clarion code thrown around in the Kitchen; in fact, the two main topics seem to be food (no surprise) and photography. Some of you know the Kitchen's photographer in residence, Leroy Schulz, from ETC (your retinas will eventually heal).

Mark Riffey thought a “Best of AK” slide show that illustrates the wide representation of Clarion programmers would be a good idea, and Leroy agreed to curate the submissions. So Leroy is soliciting digital images from Clarion photographers, and will in fact take submissions from the wider Clarion world; you don't need to be present in the kitchen. Also the best photos will be assembled into a slide show. A few submissions are already up at www.frostbytes.ca/gallery/ak2005.

These are the submission guidelines, from Leroy:

1. Please include image information including:
    1. Title.
    2. Location.
    3. Approximate date when the photo was shot.
    4. Short (2-3 sentence) description of the shot.

    I'd suggest using the Time magazine link for a good example of length of description, etc.

2. Please send high-resolution images. (I'll down-sample them if necessary.)
3. No more than five images per person, please.
4. Indicate whether you're giving me permission to modify any images (cropping, sharpening, etc.). No promises -- this is only if I find the time.

Email your photos to Leroy. Individual emails up to 5 megs should be fine.

Holiday hours and Christmas greetings!

Direct link

Posted Saturday, December 24, 2005 by Dave Harms

The Clarion Magazine office will be closed on December 26 (Boxing Day) and then open irregularly until New Year's Day. We're fortunate to have a lot of family in town for the holidays, with all of the attendant activites.

There are still a couple of articles slated for this month, including Colin Wynn on sprucing up the look of Clarion apps. Look for those later next week. The CapeSoft Profiler first look will appear in January.

We wish you and yours a Merry Christmas and happy holiday season!

[Last 25 entries] [Last 50 entries] [All entries]