Clarion Magazine

Clarion News

- » CapeSoft World Tour European Leg
- » FinalStep 2.13
- » Expert Wizards Report Application Designer
- » SetupBuilder 6.5 Build 1876
- » J-Fax and Clarion Desktop
- » Aussie DevCon Pics
- » J-Fax 1.52
- » J-Spell 1.42
- » Organize365 Source Code
- » Clarion Desktop 3.82
- » Solid Software Vacation Schedule
- » Evolution Sheet Wizard 1.0 Beta
- » StrategyOnline Forum
- » Lindersoft Releases Roadmap for SetupBuilder 7
- » SV Clarion Courses Special Offer
- » iQ-Notes Beta
- » EZRound 2.1 Released
- » J-Spell Updated
- » J-Flow 2.0
- » Johannesburg Wold Tour Event Complete
- » Clarion Developer Wanted
- » AxGC 2D Graphical ActiveX Control
- » Win A Comodo Code Signing Certificate Coupon
- » CoolButtons 1.03 Beta is now available.
- » New Whitemarsh Books
- » Translator Plus 63-06
- » CoolFrames 1.13 Beta
- » Mintoff Purchases ClarioNET
- » amazingGUI 1.1.0 Adds Jelly buttons
- » E-Books On Sale This Week Only
- » New Vista E-Book
- » Clarion MapPoint Templates 2.0.280
- » Clarion 6.3 Build 9056 Released
- » Clarion Desktop 3.80
- » Using C6 With Vista
- » SoftVelocity Driver Deals
- » GTL 6.29
- » cpTracker Source Sale
- » SetupBuilder 6.5 Developer Edition Build 1852
- » CoolButtons 1.02 Beta

Latest Subscriber Content

Pre-purchase Clarion Tips Vol 4 and save!

Clarion Tips & Techniques Volume 4 is now in final editing - **pre-purchase** before the book goes to print and **save up to $20**. Topics in our latest collection of ClarionMag gems include: browse and form handling, commonly-used Clarion embeds, writing templates, understanding threading issues, controlling printers, using SQL, getting the most out of Vista, managing source code with version control, creating DLLs, calculating dates, using finite state machines, and much much more! **More...**

Posted Thursday, May 31, 2007

Source Code Library 2007.05.31 Available

The Clarion Magazine Source Code Library has been updated to include the May source. Source code subscribers can download the January-May 2007 update from the My ClarionMag page. If you're on Vista please run Lindersoft's Clarion detection patch first.

Posted Thursday, May 31, 2007

Adding Aero Glass Effects to Clarion Apps

Vista's Aero Glass adds eye candy to Vista applications. But how do you add this to Clarion applications? Randy Rogers shows how your apps can sport the Aero Glass effect.

Posted Wednesday, May 30, 2007

Aussie DevCon: Bob Z Shows Clarion.NET

Russ Eggen reports on Bob Zaunere's closing session at the Aussie DevCon. Bob focused on the Clarion. NET product line, including data binding/browse handling, the simplicity of calling third party or . NET framework code, changes to the Clarion.NET language, web applications, PDA development, and the new debugger.

Posted Sunday, May 27, 2007

Aussie DevCon: Bob Zaunere's Opening Session

In his opening session at the Aussie DevCon in Sydney, Australia, Bob Zaunere showed the Clarion 7 IDE, including the dictionary editor, visual data designer, and some aspects of the new AppGen. He also demonstrated Clarion.NET apps for ASP.NET and mobile devices. Russ Eggen reports.

Posted Friday, May 25, 2007

Calling Google Earth From Clarion

Google Earth combines mapping and satellite imagery into a powerful, freely available earth-viewing package. And with an easily-created XML file and a single API call, you can control Google Earth from within a Clarion application. Fernando Cerini shows how it's done.

Posted Wednesday, May 23, 2007

Sending A String Between Processes

Larry Sand concludes his series on interprocess communication with a look at sending strings between processes with the InterprocessComs class.

Posted Tuesday, May 22, 2007

Vista E-Book Updated

An updated version of our newest PDF e-book: Manifests, Code Signing, and Windows Vista is now available. This collection of articles covers XP and Vista manifests, code signing, and the many issues around Vista's User Account Control (UAC). The latest release includes Jane Fleming's two recent articles on Vista's virtualization.

Posted Thursday, May 17, 2007

Coping With Vista's "Virtual" Reality, Part 2

Vista's security policies mean that folders you once wrote to in earlier versions of Windows are not necessarily writeable now, and the same holds true for some registry keys. Instead, these calls are virtualized

- » BGSoftFactory First Anniversary
- » Clarion 6.3 Build 9056 Pre-Release
- » Evolution Browse Export Tutorial

[More news]

[More Clarion 101]

Latest Free Content

- » Source Code Library 2007.05.31 Available
- » Aussie DevCon: Bob Zaunere's Opening Session
- » Vista E-Book Updated
- » Source Code Library 2007.04.30 Available

[More free articles]

Clarion Sites

Clarion Blogs

to user-specific directories and keys. Jane Fleming explains. Part 2 of 2.

Posted Monday, May 14, 2007

Synchronous Messages, SendMessage and SendMessageTimeout

Having covered asynchronous interprocess communication, Larry Sand moves on to synchronous messaging, where the sending code waits for a reply.

Posted Friday, May 11, 2007

Coping With Vista's "Virtual" Reality, Part 1

Vista's security policies mean that folders you once wrote to in earlier versions of Windows are not necessarily writeable now, and the same holds true for some registry keys. Instead, these calls are virtualized to user-specific directories and keys. Jane Fleming explains. Part 1 of 2.

Posted Tuesday, May 08, 2007

Source Code Library 2007.04.30 Available

The Clarion Magazine Source Code Library has been updated to include the April source. Source code subscribers can download the January-April 2007 update from the My ClarionMag page. If you're on Vista please run Lindersoft's Clarion detection patch first.

Posted Friday, May 04, 2007

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

Source Code

The ClarionMag Source Code Library

Clarion Magazine is more than just a great place to learn about Clarion development techniques, it's also home to a massive collection of Clarion source code. Clarion subscribers already know this, but now we've made it easier for subscribers and non-subscribers alike to find the code they need.

The Clarion Magazine Source Library is a single point download of all article source code, complete with an article cross-reference.

More info • Subscribe now

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 3 - ISBN: 0-9689553-9-8
- » 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

ISSN

Clarion Magazine's ISSN

Clarion Magazine's International Standard Serial Number (ISSN) is 1718-9942.

About ISSN

The ISSN is the standardized international code which allows the identification of any serial publication, including electronic serials, independently of its country of publication, of its language or alphabet, of its frequency, medium, etc.

Clarion Magazine

Clarion News

Search the news archive

Chicago Clarion User Group Meeting June 13

The next Chicago Clarion User Group meeting will be on Wednesday June 13th, 2007 at 6:30 p.m. in suite 270 of the Nebo Systems offices. The address is 1 South 450 Summit Ave. Suite 270, Oakbrook Terrace, IL 60181. The sign at the front of the office complex says Summit Oaks. The presentation, given by Len Sumnler, promises to be a little different. In deference to the ever growing popularity of open source tools, Len's presentation is entitled : "Snakes and MySQL and Clarion." For further information email james at nebo.com or call 630-916-8818 x 136. And please let James know if you are going to attend.

Posted Wednesday, June 06, 2007

Whitemarsh Releases Strategy for Successful Development of Business Information Systems Book

Whitemarsh announces the availability of the Strategy for Successful Development of Business Information Systems book. This book is available at a special price of $45.47 (35% off its retail price). The book contains questions and exercises at the end of every chapter suitable for job-site work-assignments to kick-start Business Information System Development efforts.

Posted Wednesday, June 06, 2007

InfoZip Wrapper Class Changes

Chris Bordeman has made a minor change to his InfoZip wrapper class. Some of the methods accepted a COMMA separated list of filespecs. Since filenames can contain commas, those methods now accept a semicolon separated list. The latest release also has a readme.txt, corrected examples, and better documentation. Listed as "InfoZip wrapper class 1.0.1" in the download section.

Posted Wednesday, June 06, 2007

A Clarion Love Song

Dave Griffiths recorded Stu Andrews' Clarion love song at the Aussie DevCon, and Stu has a link to the YouTube video. Stu's presentation notes are also available for download.

Posted Wednesday, June 06, 2007

Ingasoftplus Anniversary Sale

Ingasoftplus celebrates the sixth anniversary of its professional activity with special pricing - during June 2007 all products are 15% off.

Posted Wednesday, June 06, 2007

Huenuleufu June Specials

To celebrate the second year anniversary of Huenuleufu Development and the new family member Julian, for the whole month of June 2007 with each new purchase of Audit Pack or Support Pack you get two additional years of free upgrades and support (a.k.a. maintenance plans).

Posted Wednesday, June 06, 2007

J-Cal 1.62

J-Cal 1.62 has been released.

Posted Wednesday, June 06, 2007

Aussie DevCon Saturday Coverage

Stu Andrews has posted a summary of the Saturday content at the Aussie DevCon.

Posted Wednesday, June 06, 2007

Seal-Soft Domain Problems

Seal-Soft is having a temporary problem with its seal-soft.com domain, and the web site and email are temporarily unavailable. In the meantime you can communicate with Alexander Ivanovsky at ivanovsky at inbox.ru.

Posted Monday, June 04, 2007

CapeSoft World Tour European Leg

Anyone thinking of attending the European leg of the Capesoft World tour should book now. Richard will be finalizing numbers on Friday, June 1 and a few places are still available.

Posted Wednesday, May 30, 2007

FinalStep 2.13

New in FinalStep 2.13: Support for view form button change and view browse button setup on C6's "Legacy" chain; Minor fixes; The Read from the INI file code template was improved to read from the Styles file first.

Posted Wednesday, May 30, 2007

Expert Wizards Report Application Designer

Alpha 1.2 of Expert Wizards Report Application Designer is due for release later next week. This new version of the end user report application designer solution will incorporate a new alpha web site for alpha testers to see the lastest developments of this now very large scale development. The suite of applications has been increased to include a new report data designer prototype. Imput into the feature of the prototype will be weclomed so that this powerful new tool can cater for developers needs. Input from testers has enlarge the scope of the project to a point where its beta has been delayed while a massive number of features have been included to satistfy the large number of options required by software developers. There is likely to be a several month period of farther development while these comprehensive feature sets are integrated seamlessly into what is now a suite of applications that will make up the end user report application system.

Posted Wednesday, May 30, 2007

SetupBuilder 6.5 Build 1876

SetupBuilder 6.5 Build 1876, a maintenance release, is now available. for download. If you have a current SetupBuilder Maintenance and Support Subscription Plan the update is free of charge. You can get the latest version by selecting "Check for Updates" from within the SetupBuilder 6.5 IDE.

Posted Wednesday, May 30, 2007

J-Fax and Clarion Desktop

Anyone who buys J-Fax before the end of the month will receive a free six month license (or a six month extension) for Clarion Desktop. Clarion Desktop includes free license for Clarion Data Mapper and Organize365.

Posted Wednesday, May 30, 2007

Aussie DevCon Pics

Stu Andrews has put up some DevCon pictures. They're lightbox enabled, so click on the photo to get the full series.

Posted Thursday, May 24, 2007

J-Fax 1.52

J-Fax 1.52 is now available. The fax server app has been improved, and there are three new reports so you can print out which faxes were sent, which failed, and which are still queued. The installer and documentation have also been redone.

Posted Thursday, May 24, 2007

J-Spell 1.42

J-Spell 1.42 is now available.

Posted Thursday, May 24, 2007

Organize365 Source Code

The latest version of the Organize365 source code has been released. The next build should follow towards the

end of next week, and will hopefully include the newsgroup reader and the new email editor controls.
Posted Thursday, May 24, 2007

Clarion Desktop 3.82

Clarion Desktop version 3.82 is now available. You can now delete products from your Password Manager (in the past the products remained in the cache table and reappeared every time you checked for updates), some html characters (quotation marks etc) were appearing mangled in the news blocks, and you can now only run one instance of Desktop - solving that "this port is already used" error that some users reported.
Posted Thursday, May 24, 2007

Solid Software Vacation Schedule

The Solid Software office will be closed until June 4th. Any support requests will be dealt with thereafter.
Posted Thursday, May 24, 2007

Evolution Sheet Wizard 1.0 Beta

Evolution Sheet Wizard 1.0 Beta is a tool for building wizards. This is a control template with full source code. Demo available.
Posted Thursday, May 24, 2007

StrategyOnline Forum

StrategyOnline has set up a product support forum. There are also links from the various product pages that will take you to the specific area on the forum for that product.
Posted Thursday, May 24, 2007

Lindersoft Releases Roadmap for SetupBuilder 7

The next milestone release (SetupBuilder 7.0 Beta) is expected to be beleased in Q3 2007. SetupBuilder 7 will be free for customers with a current SetupBuilder maintenance subscription. Key SetupBuilder 7 MSI Features include: Native MSI support; SetupScript Bootstrapper compiles a full featured generic installer that lets you package multiple .MSI files into a single-file or multiple-volume native Windows .EXE; SetupScript Custom Actions to call SetupScript for MSI functions as custom actions from the .MSI; Configurable Merge Module (.MSM) support; MSI API calling; MSI Web Update. SetupBuilder 7.0 Beta is expected to be released in Q3 2007. Pricing for the current SetupBuilder 6 installation software begins at US$179 for Professional Edition. The best-of-breed Developer Edition starts at US$249. A trial version is available.
Posted Thursday, May 24, 2007

SV Clarion Courses Special Offer

Until June 15, 2007 you can get 50% off the following Clarion Training On Demand courses offered by SoftVelocity: Course III, Expert Programming in Clarion; Course IV, Master's Language Series; Course V,

Template Language Series; Course VII, SQL Programming Series; Course VIII, Programming with the in-Memory Driver 2.0 Templates and Classes.
Posted Thursday, May 24, 2007

iQ-Notes Beta

A beta release of iQ-Notes available. This free release is written in Clarion 6 and is a replacement for StickyNotes. iQ-Notes is much different than StickyNotes with many enhancements. The most significant enhancement is that iQ-Notes synchronizes with an FTP server. If you have more than one computer the "Post-It" notes will stay in sync between your computers. In addition, there are at least 20 enhancements including the new Manage Notes screen, flags, encryption, more colors, rollback, new printing options, saving to a text file and more. iQ-Notes also supports VISTA when UAC is turned on, storing data in the ProgramData folder.
Posted Monday, May 14, 2007

EZRound 2.1 Released

LANSRAD announces the release of EZRound 2.1 software for Microsoft Windows. Version 2.1 of the software has a Windows Vista aware program installer. It includes new code generation templates that allow HTML containers created by EZRound to pass W3C XHTML validation. There were also other new enhancements and minor bug fixes. This is a free upgrade for all registered EZRound users. Current users can use the "Check for Updates feature" on the EZRound help menu to obtain the new version. Users moving to a new Windows Vista computer should download the new installer and do a fresh install with it (the previous installer is not a Vista aware installer).
Posted Monday, May 14, 2007

J-Spell Updated

There was a problem using J-Spell and xPathManager in the same APP; this has now been fixed. Also, the installer now supports Clarion 7.0.1923.
Posted Monday, May 14, 2007

J-Flow 2.0

J-Flow is a template that enables you to use this the Lassalle Technologies AddFlow flow diagram control in Clarion applications.
Posted Monday, May 14, 2007

Johannesburg Wold Tour Event Complete

The Johannesburg event has wrapped up, and time is running out for registrations for the Australian, USA and UK events. Registrations have to close on Wednesday, May 16 2007.
Posted Monday, May 14, 2007

Clarion Developer Wanted

A Sydney Australia based company with a team of five existing Clarion developers is looking to appoint a new clarion programmer. This will be a full time onsite position based in the CBD. The successful applicant will have one to three (or more) years of programming experience in Clarion for Windows good
business presentation good spoken and written English skills analysis / database design skills would be nice. Knowledge of SQL script, MSSQL, XML is a distinct advantage. Any work background in the area of General Ledger data integrations would be an advantage. Senior contractors are invited to an make an application. This position will start as a mid level programming role and will progress based on performance. Initial appointment will be three months (with notice period of one day). After three months (on reappointment) it will revert to a rolling 4 week contract. The role has long term potential. This position will be onsite in Circular Quay Sydney. Please call Richard Bryce for details on 0403 892880 (Australia). No emails in the first instance. This is a full time office based role. Contractors with shelf companies that are fully insured are ok. No agencies.

Posted Monday, May 14, 2007

Clarion Magazine

Clarion Tips & Techniques Volume 4

At over 650 pages, Clarion Tips & Techniques Volume 4 is another blockbuster brimming with terrific articles from Clarion Magazine.

Clarion™
Tips &
Techniques
Volume 4
Time-saving, money-saving, sanity-saving gems from the pages of Clarion Magazine.
Edited by David Harms

Now in final editing - pre-purchase and save $$!

This book is now in final editing, and we expect it to ship in late June or early July. The list price is US$79.95 - right now you can buy this book for just $64.95, or $59.95 if you have a current Clarion Magazine subscription (as a subscriber you also get big discounts on e-books!). When the book goes to the printer, the price goes up!

Pre-purchase now in the Clarion Magazine store.

Contents

The draft table of contents (subject to minor changes) is now available. Topics covered in this book include:

- Browses & Forms
  - Internationalization standards
  - Dynamic listbox formatting
  - Edit-In-Place
  - Hot fields
  - Recursive inserts and updates
  - Beautifying Clarion apps
  - Automatically closing windows
  - Commonly used Clarion embeds
- Templates
  - Multiple lookups
  - Record status controls
  - A template debugger
  - Class wrappers
- Threading
  - Global variables
  - Critical sections
  - Background processes

  - Controlling START
- Reports
  - Sending printer codes
  - Page loaded trees
  - Printing unknown queue fields
  - Printing directly to the printer
  - Printing directly to USB printers
  - Printing a "No records" report
- Databases
  - Mimer
  - The SQL Advanced settings
  - Creating SQL from XML with XSLT
  - External business rules
  - The in-memory driver
  - Using SQLIdentity
  - Multi-user primary keys
  - Embedded SQLite
  - PROP:SQL
- Windows Vista
  - Encryption and application signing
  - Manifests
  - Getting ahead with Vista and Office 2007
  - Vista-compliant INI files
  - Running C6 on Vista
- Version Control
  - CVS and WinCVS
  - Using CVS/WinCVS with Clarion 6.x
  - Understanding the C6 version control interface
  - Using MS Visual Source Safe with Clarion
- DLLs
  - Eliminating circular DLL calls
  - Smart DLL loading
  - DLLs and reusable code
  - Generic DLLs
  - Hand coding export files
- Tips & Techniques
  - Encrypting data
  - App shutdown options
  - Accessing private class data
  - Sorting queues
  - Displaying queues
  - Date calculations
  - Adding arrays to generic queues
  - Customizing deep assigns
  - Using finite state machines

- Using metadata
- Providing good customer service

Clarion Magazine

Adding Aero Glass Effects to Clarion Apps

by Randy Rogers

Published 2007-05-30

A recent Microsoft Developer Network magazine contained an article on creating special effects with the Desktop Window Manager. The Desktop Window Manager (DWM) is the new Vista interface that manages how the windows of running applications are merged together and rendered on the desktop. It is this part of Vista that provides the Aero Glass effect. The article got me wondering how easy or difficult it might be to get the Aero Glass look in a Clarion 6 application.

I first visited the url printed in the article for the available code download only to discover that the file was not, at that time, available; it is currently available. Google also identified a couple of good articles, one by Tim Sneath and another by Kenny Kerr. The official Microsoft documentation can be found on MSDN. I also had to download the Microsoft Windows Software Developer Kit 6.0 to get the header files I needed to create the function prototypes and constants contained in the dwmapi.inc file.

I spent a lot of time trying to mimic code that was presented in the articles and managed to get some very strange results. After much trial and error I finally got an example to work pretty much like I wanted. That example is presented in this article.

In my example the main DWM API call is the DwmExtendFrameIntoClientArea function, which is used to extend the window frame behind the client area of the window so the background shows through. The frame can be extended on any of the four sides of the client area. It is even possible to cover the entire client area by specifying -1 for any of the margins. The Vista Windows Media player uses this to good effect for the controls at the bottom of the player. Using Clarion 6 to achieve some of these same effects is not very easy because the runtime library uses older GDI API functions. To render most things properly it is necessary to use GDI+ API functions that support alpha blending. GDI+ is beyond the scope of this article, but I was able to get toolbar buttons to render properly without too much effort.

This example is a simple application frame with a toolbar and two buttons, one to exit the program and one to start an mdi child window.

Constants and globals

I include the dwmapi.inc file that contains function prototytpes, group structures and equate constants for the DWM API. Also in the file is the OSVersionInfo needed for version checking and a few global variables for the application.

```
  PROGRAM
  INCLUDE('dwmapi.inc'),ONCE       !Include Desktop Window Manager
S_OK                EQUATE(0)
TCHAR               EQUATE(BYTE)
WPARAM              EQUATE(SIGNED)
LPARAM              EQUATE(LONG)
GCL_WNDPROC         EQUATE(-24)
WS_EX_LAYERED       EQUATE(00080000h)

! Layered Window Attribute flags
LWA_COLORKEY        EQUATE(000000001h)
LWA_ALPHA           EQUATE(000000002h)
!
OSVERSIONINFO       GROUP,TYPE
dwOSVersionInfoSize   DWORD
dwMajorVersion        DWORD
dwMinorVersion        DWORD
dwBuildNumber         DWORD
dwPlatformId          DWORD
szCSDVersion          TCHAR,DIM(128)
                    END
bUseGlass           BOOL(TRUE)
glassMargins        LIKE(MARGINS)
ClaButton:origWndProc   LONG
```

The MAP sets out the prototypes for the six procedures in the program, includes the svapifnc.inc API function declarations, and a few additional windows API prototypes needed by the program and not defined elsewhere. A lot of windows APIs are linked into and exported by the Clarion RTL but there are occasions when it is necessary to use a newer API that is not exported by the RTL.

You will want to access the APIs using the LoadLibrary and GetProcAddress functions. I used this approach for both the SetLayeredWindowAttributes and the DWM functions. If you look closely at the prototype for the SetLayeredWndowAttributes you will notice the addition of the DLL attribute, and, following the MAP, a long variable used to hold the procedure entry point. Similar declarations are used in dwmapi.inc for the DWM functions. The code to do the actual library initialization is in the example’s InitLibrary procedure.

```
  MAP
    Main(),BOOL,PROC
    GlassForm()
    NewWindow()
    InitLibrary(),BOOL
    ClaButton:wndProc(HWND hWnd, UINT wMsg, |
      WPARAM wParam, LPARAM lParam),LONG,PASCAL
    FixButtonBorder(HWND hWnd)
    INCLUDE('svapifnc.inc'),ONCE
    MODULE('kernel32.dll')
      GetClassLong(HWND hWnd, LONG nIndex),DWORD,PASCAL,|
        NAME('GetClassLongA')
      GetVersionEx(*OSVERSIONINFO osvi),BOOL,PASCAL,|
        PROC,RAW,NAME('GetVersionExA')
      SetLayeredWindowAttributes(HWND hWnd, COLORREF crKey, |
        BYTE bAlpha,DWORD dwFlags),BOOL,PASCAL,PROC,|
        DLL(__SetLayeredWindowAttributes)
    END
  END


__SetLayeredWindowAttributes LONG,AUTO,|
    NAME('SetLayeredWindowAttributes')
```

The Main procedure

When the application starts up, the first call runs the Main procedure:

```
  CODE
  Main()
  RETURN
```

Main checks the OS version and set the bUseGlass variable. I can only use the DWM glass effect if the OS is Vista (6). Main then calls the GlassForm procedure.

```
Main         PROCEDURE
osvi          LIKE(OSVERSIONINFO)
hr           HRESULT
hDC            HDC
  CODE
  !Check to make sure that we are running in Vista or later
  osvi.dwOSVersionInfoSize = SIZE(OSVERSIONINFO)
  IF GetVersionEx(osvi)
   bUseGlass = CHOOSE(osvi.dwMajorVersion < 6,FALSE,TRUE)
   GlassForm()
  END
  RETURN TRUE
```

The GlassForm procedure

GlassForm is the application frame with a toolbar and a couple of buttons.

```
GlassForm      PROCEDURE
hr             HRESULT(S_OK)
bReturn        BOOL(FALSE)
fEnabled       BOOL(TRUE)
hwnd           LONG
bCheckBox      BOOL

Window APPLICATION('Vista Glass Demo'),AT(,,400,300),|
         FONT('Verdana',10,,FONT:regular,CHARSET:ANSI), |
         CENTER,ICON('windows.ico'),SYSTEM,RESIZE
       TOOLBAR,AT(0,0,,24),USE(?ToolBar),COLOR(0FEFCFCH)
         BUTTON,AT(2,2,20,20),USE(?Exit:Button),|
           ICON('Shutdown.ico'),STD(STD:Close)
         BUTTON,AT(25,2,20,20),USE(?Window:Button),ICON('windows.ico')
       END
     END
  CODE
```

If I can use the DWM, I initialize the library

```
  IF bUseGlass = TRUE
    bUseGlass = InitLibrary()
  END
  OPEN(Window)
```

The window is now open. If I can use the glass effect, now is the time to do it.

First I call DwmIsCompositionEnabled. When composition is enabled windows no longer draw directly on the screen surface as in previous versions of Windows. Instead, they are redirected to off screen surfaces in video memory and then rendered into a desktop image by the Desktop Window Manager. This allows the DWM to provide the Aero Glass effect. The code for extending the glass frame is only executed if composition is enabled.

```
IF bUseGlass = TRUE
  hr = DwmIsCompositionEnabled(fEnabled)
  IF hr = S_OK AND fEnabled
```

The DWM uses black as the transparency color so I set the toolbar color to black since I want it to get the Aero Glass effect and make the buttons transparent.

```
?ToolBar{PROP:COLOR} = COLOR:BLACK
?Exit:Button{PROP:TRN} = TRUE
?Window:Button{PROP:TRN} = TRUE
```

By default Clarion does not create windows with the ws_ex_layered attribute which is necessary for the Aero Glass effect. The SetWindowLong and GetWindowLong functions are used to add the required attribute and then the SetLayeredWindowAttributes function is used to make the window opaque. A value of 255 for the third parameter means totally opaque and a value of 0 means transparent. The lwa_alpha flag tells the function that the alpha channel value is to be set.

```
!get the window handle
hWnd = Window{PROP:Handle}
!make sure its a layered window
SetWindowLong(hWnd, GWL_EXSTYLE, |
  BOR(GetWindowLong(hWnd, GWL_EXSTYLE) , WS_EX_LAYERED))
!make it opaque
SetLayeredWindowAttributes(hWnd, 0, 255, LWA_ALPHA)
```

The DwmExtendFrameIntoClientArea uses a margins structure to specify how much to extend the frame into the client area. MARGINS consists of four long values, one for each edge of the frame. A value of -1 for any edge will cause the entire client area to be glassed. I want to glass over the toolbar at the top of the window, so I get the height of the toolbar in pixels, put that value in the glassMargins. cyTopHeight variable, set the other edges to zero and call the DwmExtendFrameIntoClientArea function.

```
!set the glass margins
Window{PROP:Pixels} = TRUE
glassMargins.cyTopHeight    = ?ToolBar{PROP:Height}
Window{PROP:Pixels} = TRUE
glassMargins.cxLeftWidth    = 0
glassMargins.cxRightWidth   = 0
```

```
      glassMargins.cyBottomHeight = 0
      !extend the glass into the client area
      DwmExtendFrameIntoClientArea(hwnd, glassMargins)
```

There is one small detail left to take care of and that is the rendering of the buttons on the glass. Without this next step the buttons will appear on the glass with white borders around them. The button wndprocs need to be subclassed to allow the program to fix up the buttons so they look better on the glass. All buttons use the same wndproc, so it is only necessary to save one origWndProc and then redirect the others to the program replacement.

```
      !subclass the button wndprocs
      ClaButton:origWndProc = ?Exit:Button{PROP:WndProc}
      ?Exit:Button{PROP:WndProc} = ADDRESS(ClaButton:wndProc)
      ?Window:Button{PROP:WndProc} = ADDRESS(ClaButton:wndProc)
    END
  END
```

Last comes the code process the window events. With any luck the toolbar will look ok on XP and get a nice glassed appearance on Vista.

```
  !process messages
  ACCEPT
    CASE EVENT()
    OF EVENT:Accepted
      CASE ACCEPTED()
      OF ?Window:Button
        START(NewWindow,25000)
      END
    END
  END
  !Close the window
  CLOSE(Window)
  RETURN
```

The NewWindow procedure is nothing special, just an MDI window to give one of the toolbar buttons something to do.

```
NewWindow PROCEDURE

Window WINDOW('New Window'),AT(,,160,50),SYSTEM,GRAY,|
       DOUBLE,MDI,ICON('windows.ico'),|
           FONT('Verdana',10,,FONT:regular,CHARSET:ANSI)
      STRING('Vista Glass Demo!'),AT(28,16),USE(?String1),|
         FONT('Veranda',18,,FONT:bold)
     END
    CODE
    OPEN(Window)
    ACCEPT
    END
    CLOSE(Window)
```

Aero Glass source code

I've written several functions to make it easier to use the Aero Glass effect in Clarion apps. These include InitLibrary, ClaButton:wndProc, and FixButtonBorder.

Initializing the library

The InitLibrary procedure dynamically loads and finds procedures from the dwmapi and user32 DLLs. This is a simplified version hooking only the functions actually used in the article. The downloadable source version of this procedure hooks all the dwmapi functions.

```
InitLibrary PROCEDURE   !,BOOL

bReturn        BOOL(FALSE)
szDwmApi       CSTRING('dwmapi.dll')
szUser32       CSTRING('user32.dll')
hDwmApi        HANDLE
hUser32        HANDLE
szProcName     CSTRING(65)
  CODE
 ! load DWMAPI.DLL
 hDwmApi = LoadLibrary(szDwmApi)
 IF (hDwmApi <> 0)
   ! locate functions in DWMAPI.DLL
   szProcName = 'DwmExtendFrameIntoClientArea'
   __DwmExtendFrameIntoClientArea = |
     GetProcAddress(hDwmApi, szProcName)
   szProcName = 'DwmIsCompositionEnabled'
```

```
     __DwmIsCompositionEnabled = |
       GetProcAddress(hDwmApi, szProcName)
   END
   ! load USER32.DLL
   hUser32 = LoadLibrary(szUser32)
   IF (hUser32 <> 0)
     ! locate functions in USER32.DLL
     szProcName = 'SetLayeredWindowAttributes'
     __SetLayeredWindowAttributes = |
       GetProcAddress(hUser32, szProcName)
   END
   !Check for Errors
   IF __DwmExtendFrameIntoClientArea = 0        OR |
     __DwmIsCompositionEnabled = 0              OR |
     __SetLayeredWindowAttributes = 0
     bReturn = FALSE
   ELSE
     bReturn = TRUE
   END
   RETURN bReturn
```

Fixing up the buttons

ClaButton:wndProc is the subclassed wndproc for the buttons. After the RTL draws the buttons it is necessary to replace the button frame with black lines so that the DWM will render them as glass. When the buttons receive wm_setfocus or wm_paint messages they call the FixButtonBorder utility procedure.

```
ClaButton:wndProc PROCEDURE(HWND hWnd, UINT wMsg, |
                    WPARAM wParam, LPARAM lParam)



retVal      LONG
  CODE
 !let clarion rtl handle the message
 retVal = CallWindowProc(ClaButton:origWndProc,|
   hWnd,wMsg,wParam,lParam)
 CASE wMsg
   OF WM_SETFOCUS       !fix borders when button gets focus
     FixButtonBorder(hWnd)
```

```
    OF WM_PAINT          !fix borders when button is painted
      FixButtonBorder(hWnd)
  END
  RETURN RetVal
```

Fixing the button border

FixButtonBorder is a little utility procedure that uses a black pen to draw a frame around the button window.

```
FixButtonBorder   PROCEDURE(HWND hWnd)



rect        LIKE(_RECT_)
hDC         HDC
hRegion     HRGN
hBrush      HBRUSH
   CODE
  IF GetClientRect(hwnd, rect)
     hRegion = CreateRectRgn(rect.left, rect.top, |
       rect.right, rect.bottom)
     hBrush = GetStockObject(BLACK_BRUSH)
     hDC = GetDC(hWnd)
     FrameRgn(hDC, hRegion, hBrush, 3, 3)
     DeleteObject(hRegion)
     ReleaseDC(hWnd,hDC)
  END
  RETURN retVal
```

Screen shots

So what does it look like? Figure 1 shows the "before" image of the application running on XP.

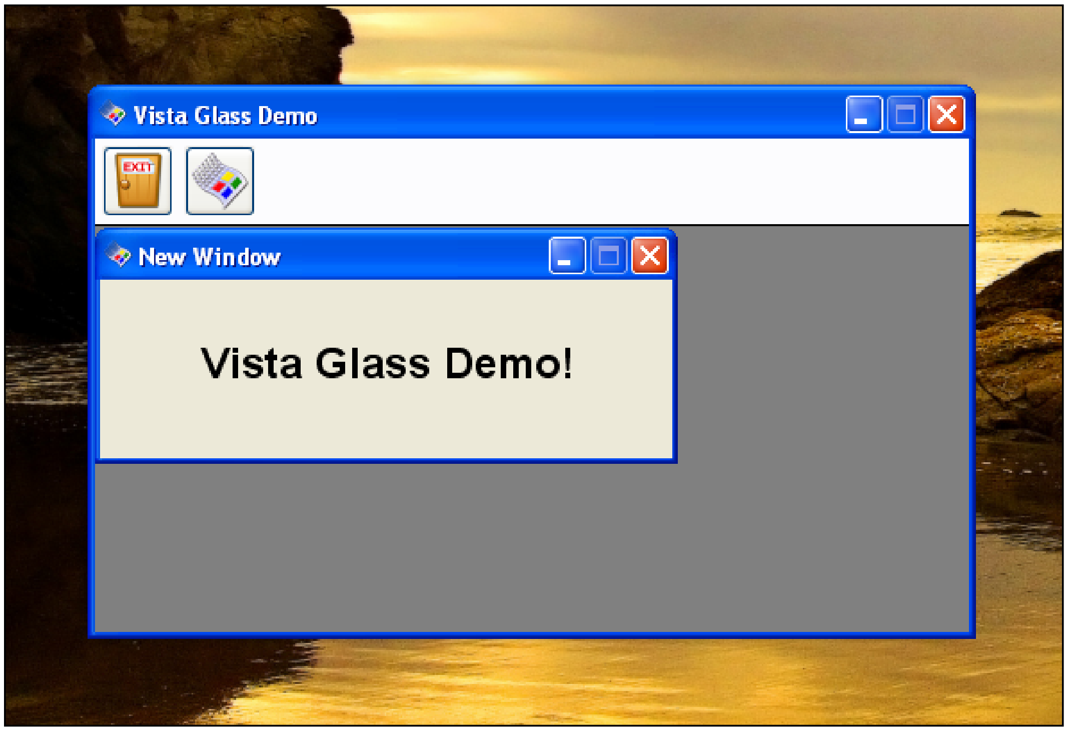


Figure 1. The app running on XP

And Figure 2 is an image of the application running on Vista.

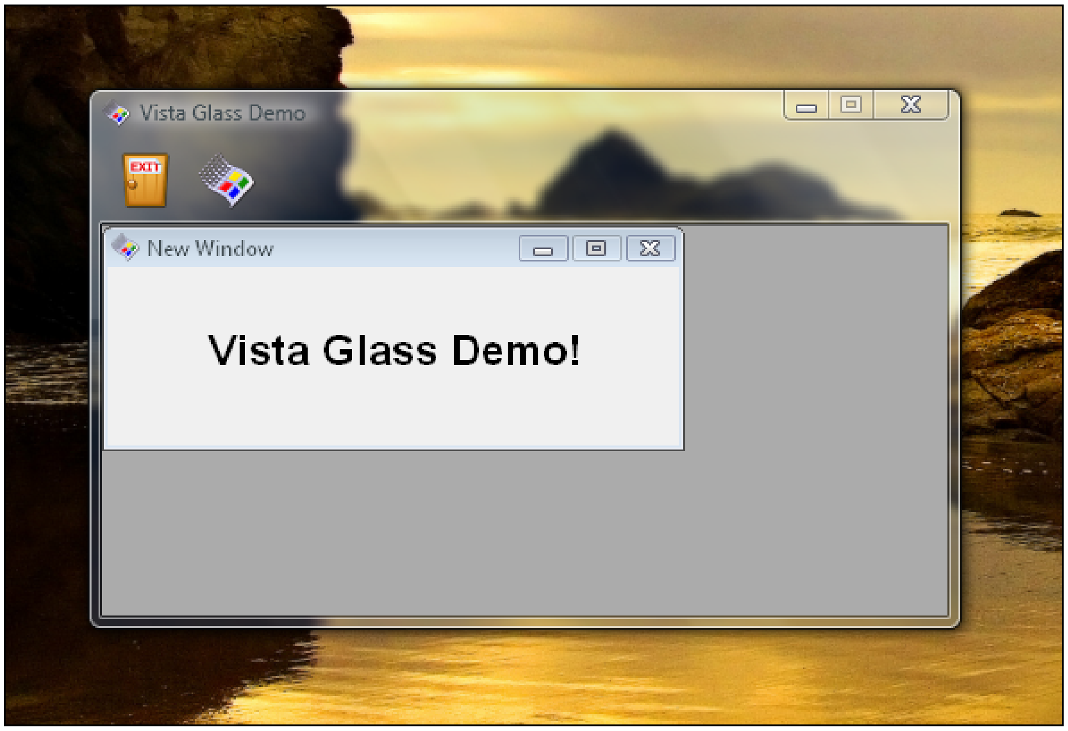


Figure 2. The app running on Vista

Summary

Clarion doesn’t support the Aero Glass effect natively, but you can still get there with a little judicious use of the Windows API. For compatibility with older versions of Windows you should dynamically load the necessary DLLs.

Download the source

Randy Rogers is a data processing professional with over 35 years of experience in a wide variety of industries including accounting, municipal government, insurance, printing, and pharmacoeconomics. He has a degree in Mathematics from Florida State University and is the president of Keystone Computer Resources. Randy is the author of ClassViewer, a utility for browsing the Clarion class hierarchies. He is also the creator of NetTools, Queue Edit-in-Place, and Screen Capture Tools for Clarion application developers.

Reader Comments

Posted on Wednesday, May 30, 2007 by ChrisRussell

Brilliant! Anything is possible with the good ol' APIs, and Clarion, of course! Great article Randy!

Add a comment

Clarion Magazine

Aussie DevCon: Bob Z Shows Clarion.NET

by Russell Eggen

Published 2007-05-27

Bob Zaunere closed out the Aussie DevCon with a presentation focused on the Clarion.NET product line. Key areas included data binding/browse handling, the simplicity of calling third party or .NET framework code, changes to the Clarion.NET language, web applications, PDA development, and the new debugger.

Data binding

Most .NET controls support data binding to some extent. All of the important entities, including FILE, RECORD, QUEUE and GROUP, can be bound to any .Net control that supports data binding. The data binding will be automatically handled by the templates, but a hand coder still has complete control.

Bob showed an example of direct binding to a file, a record and a queue using the .Net DataGrid, DataGridView, ComboBox, TextBox and BindingNavigator. The running app showed a browseless edit form (Access style), with a .NET navigator control that used tabs to navigate to previous and next records. Thinks of tabs that acted more like buttons.

Bob then proceeded to show how this was done in source. One thing I noticed was the assignment operator := which is a smart operator (my term). This operator is the same as dest = source or dest &= source, which means it will figure out what you need and do a copy for variables and an address assignment for reference variables, as the situation dictates. You don't have to use this operator; you can still use &= on the .NET side and the compiler will know what you want.

Paged browses

.NET does not come with a list control that does the page loading Clarion developers have come to expect. Clarion.NET list controls, however, *do* support page loading.

The next demo showed two different list controls, one a list the other a date grid. Both were bound to the data record and both stayed in sync.

Language issues

All .NET code must be declared as belonging to a namespace. If you don't declare a namespace in your own Clarion.NET code, Clarion will add a default one for you.

One important new feature in Clarion.NET is the new TYPE attribute for a data file. This means you can make as many instances of a file you need, and most likely that will make alias definitions obsolete. For example, if the table is PEOPLE, the code could look like:

```
SomePeopleInstance &= People
```

The whole point of this section of the demo was to show data binding, but doing it the "Clarion way", which appears fairly straightforward. Bob went through the code fairly quickly, so I do not have more examples to include in this write-up. But the core idea seems to be that Clarion developers don’t have to do anything new in .NET to make their code work, especially with data binding.

Using .NET controls

Bob stated that you can use any control available for .NET. By way of example he showed a WebBrowser control and a link control (for URLs); clicking on the link resulted in the web page displaying in the browser. The web page was launched with this code, which functions much like ShellExecute in Win32:

```
System.Diagnostics.Process.Start('http://www.aussiedevcon.com')
```

Bob also showed a nice date/time picker control, a slider, and even a property grid (of the kind used in the C7/Clarion.NET IDE). He bound the property grid to a form, and changing something like the background color was instant, all at run-time. The usefulness of that may not be immediately apparent, but you could use a similar approach to let users specify date formats, currency, decimal pictures, etc.

Third party .NET controls

Bob then showed how easy it is to add your own controls (such as those from third party vendors). You can make your own custom controls too. You can place controls in the Global Assembly Cache (GAC) if you wish, where they can be used by other applications. You can also make use of any controls already in the GAC. As well, the GAC can hold multiple versions of a single control, which means your customers won't have code breakage due to version mismatches. You can use GAC controls from within the IDE; they appear on the toolbox and you can add them to your windows or reports. One comment from the audience was that the GAC replaces the scary COM controls and Bob agreed.

From what Bob showed, using any .Net control is very easy. I'm thinking they are similar to a control template in C6; you can group a series of controls as a single component and add them all via a single action.

Someone asked if Clarion could run under Visual Studio. Bob mentioned that VS support is planned, but certain events need to happen first, like releasing Clarion 7.

On the second half of his presentation, Bob showed that you can navigate anywhere in the IDE without a mouse. So keyboard-intensive developers should be quite happy about that.

Data browser

Next was the database browser which is the data grid replacement for the dictionary's file browser. There is a nice filter where you can pick a field, add a comparison operator and you in essence have a simple QBE. There is also a free form expression control which has auto-complete. The free form control has the ability to add additional expressions like AND and OR. The columns can be repositioned by drag and drop, and the IDE remembers these settings between sessions.

Compact framework

The .NET Compact Framework was next. This framework uses a different runtime than WinForms. To test compact apps you need an actual device or you have to use the built-in emulator. In Bob's example the emulator looked like a mobile phone or a Pocket-PC PDA. The demo showed a simple list box with edit, delete and insert buttons, all done with Clarion.NET code. In designer mode the window looks like the device itself, and by that I mean it's photorealistic. Bob says these are available from MS as a download. When you change a device while in the designer, the image changes to reflect the PDA chosen.

There are plans to supply a Topspeed data driver for compact devices in the future.

ASP.NET

Bob compiled the School_ASP.NET example with the ASP compiler and ran it in a browser. This app looked much like the example shipped with Clarion, albeit slightly webbed. And unlike past web solutions, it looked far better and used buttons, drop down menus, etc. It seemed to be a nice, responsive application. We looked at the generated ASP.NET source code, and also viewed the app in the Designer where it appeared just like it did in the browser. This is really nice - no more generating code via templates and then running it to see what your edits looks like. Exciting stuff! Bob says CSS support is there as well.

The ASP.NET application's user interface is aspx code, and behind the scenes is aspx.cln code, which is Clarion code. The ASPX code looks like a whole lot of HTML code, all of which is controlled by the designer. It's good enough to give to a Dreamweaver guy if you want. The advantage here is that the web guys can't break your business logic since that code is separate.

One of my biggest complaints with many web sites is data validation, which is seldom instant. Usually you have to submit the form and if there are multiple errors you'll only be told of the first one. Bob showed that you can have validation code on the form itself, for example a drop down list, so it's easy to ensure users do not enter or use invalid data. I like that approach a lot.

Clarion.NET Enterprise will have support for all three platforms – WinForms (desktop apps), ASP.NET (web), and Compact Framework. Professional will support only desktop and not ASP.NET or compact apps. The templates for ASP.NET are new; they are not legacy, and they are not ABC. The Clarion.

NET template set is also new, but SV has no plans to abandon Win32 and the ABC chain.

.NET debugger

The final demo was the .NET debugger. My first impression is that the debugger’s UI is vastly improved over the current Win32 debugger. Bob set breakpoints *before* the debugger was launched, then stepped through code. The debugger showed local data in nice tree structures, with data types and values, including queue data. This is going to be a *very* nice tool!

Bob was asked if he felt these product releases will happen this year and he indicated that will happen. Bob also answered the question as to how much the Clarion.NET platform will cost, and he indicated it should be about what the EE/PE editions cost now. But existing Clarion customers (and subscribers) will get a substantially discounted rate. No figures were quoted.

I have to close this by saying that I am not doing Bob's presentation much justice and I am sure I missed some key points as most of it was visual; you will have to forgive me for trying to describe in a sentence or two what we were seeing. Talking to the attendees afterwards, all agreed it was an impressive demonstration of what is to come. I was optimistic before about Clarion, but after seeing today's presentation (and forgive the gush) I am enthusiastic about these upcoming products. I feel once they do see the light of day, many in our community will feel the same.

Russ Eggen has been using Clarion since 1986. Until about 1996, he was using it for business applications, mostly accounting programs. Afterwards he joined Topspeed as a consultant, and later as an instructor. He was a founding member of SoftVelocity when that company formed from Topspeed in May 2000. He left SoftVelocity in January 2001 and now works for his own company, RadFusion Inc. He still teaches and lectures, and is currently working on a new book and setting up a local Clarion classroom. Russ enjoys flying, scuba, and applied philosophy, and with great effort you might coax him into political discussions.

Reader Comments

Add a comment

Clarion Magazine

Aussie DevCon: Bob Zaunere's Opening Session

Russ

by Russell Eggen

Published 2007-05-25

Bob Zaunere's opening session at the 2007 Aussie DevCon in Sydney, Australia covered aspects of the Clarion 7 and Clarion.NET IDE, including the new dictionary editor and the visual data designer. Bob also offered a preview of the new AppGen, and showed a Clarion ASP.NET version of the School app as well as Clarion code running on a mobile device emulator.

The new IDE

Bob started with the working bits of the IDE, showing the various docking toolbars and how to collapse them and/or dock anywhere you want. Next he explained solution files, which are the C7/Clarion.NET equivalent to projects (although a solution can also be a group of projects). Bob demonstrated a solution and showed the various settings, including setup options.

What was seen for the first time was the application options, which the alpha team does not yet have. These look exactly the same as the dialogs in C6, so Bob did not spend much time there since there was nothing really new.

Next Bob went over the redirection file and some of the differences in the new IDE, mainly that it allows you to switch between multiple Clarion versions going back to Clarion 4.

He mentioned that there are two threads in the IDE, one for the GUI and one for the background tasks like parsing code. The background thread runs concurrently and handles the loading processes.

After opening a few source files, Bob demonstrated bookmarks and mentioned they are saved between sessions; if you collapse blocks of code in the editor (code folding), the collapse states are also saved between session.

Clarion.NET and WinForm

Bob then showed a quick demo of a WinForm application, which was the same School app shipped with the current version of Clarion; there will be more details about this in the coming days. He opened multiple browses and showed that you can dock these windows in the same manner as the toolboxes in the IDE, since the application code uses the same docking library as the IDE.

Next Bob showed the WinForm designer. This designer has the same properties toolbox as the C7 designer so you may access any window and control attributes. Clarion.NET has an additional event toolbox. Bob did not go into any details – that's for a later date. Bob then contrasted the WinForm designer with the C7 window designer.

He then demonstrated the source editor's search and replace dialog (which reminds me of TextPad's search and replace). A question came up about undo capability. Bob mentioned he could drastically change a window and then save it, come back and decide he did not like it and undo it to the way he found it. The search dialog also does regex (regular expression) searches.

One of the new designer features is snap lines. You can align various controls with each other based on the top, bottom, left or right sides, something I find considerably faster than C6's alignment functionality.

Bob made the point that the window designers for C7 and Clarion.NET are pretty much the same **-** if you learn one side, you pretty much have the other side learned. The same holds true for reports.

Clarion.NET web apps

Next Bob opened an ASP.NET solution. The window designer for ASP.NET applications is the same, as far as the developer tool set is concerned. In other words, each running application (C7, WinForms, ASP.NET) looked different, but the IDE tools were the same.

The ASP.NET version of the School app, running on IE7, looked like a web app but behaved like a desktop app. I think it is the best looking web app out of the box we've ever seen with Clarion tools.

Mobile apps

Bob then showed something I've never seen before: the device emulator. This looked like an on-screen image of a mobile phone, and the application running in that emulator was Clarion code running on a mobile device. Another reason for .NET code! This got a good round of applause.

Clarion.NET is a full fledged .NET language which means it can work with any other .NET language. They say "assembly", we say "DLL". But a C# guy can get your stuff, he can give you his stuff and both libraries will work. Just use each other's namespaces and that is all you need.

Dictionary editor

Next came the dictionary demo. The dictionary is not completely done, and by that Bob specifically meant the user interface. He opened two dictionaries just to show that you can open more than one, but you are not limited to only a couple. The dictionary editor looked similar to C6, yet better and cleaner. You can see keys and relations at the same time now, using the quick view. Double clicking opens the various editors to get to the details and you can dock these editors; as with the rest of the IDE, you can set up the various dictionary windows the way you like. Drag and drop was asked about (drag from one DCT to another) and Bob said it was not there, but it was good suggestion.

I asked if it was possible to have multiple dictionaries per solution and Bob replied that it was not going to happen soon, but there are plans to use virtual dictionaries down the line.

Data Diagrammer

Next up was Data Modeller's replacement, the Data Diagrammer (for Enterprise versions of C7 and Clarion.NET only). My first impression is that Data Diagrammer looks like Visio. It is a vast improvement over Data Modeller. You can do all your dictionary work in Data Diagrammer if you so choose. I am thinking I might, as it looks quite nice. Resizing, scrolling, defining relationships, all the expected features are there. Move a table, and the relationship lines move with table, *smoothly*! Colors, fonts, etc are customizable, and you can print to PDF or save to an image (BMP for example). You can create a note on the diagram and attach it to a table.

AppGen

Bob showed a preview of th new AppGen but refused to go deeper since there is more work to be done. But it’s nice to see this and we are the first outside SoftVelocity’s offices to get a look at it. There is at least some familiar look about the new AppGen - think of the procedure property dialog with MakeOver applied. The point was to show there is work going on there, but it’s not quite ready for a full demo.

Eggen has been using Clarion since 1986. Until about 1996, he was using it for business applications, mostly accounting programs. Afterwards he joined Topspeed as a consultant, and later as an instructor. He was a founding member of SoftVelocity when that company formed from Topspeed in May 2000. He left SoftVelocity in January 2001 and now works for his own company, RadFusion Inc. He still teaches and lectures, and is currently working on a new book and setting up a local Clarion classroom. Russ enjoys flying, scuba, and applied philosophy, and with great effort you might coax him into political discussions.

Reader Comments

Posted on Friday, May 25, 2007 by Bob Campbell

This was a great article. Thanks for flying down under and reporting to us so quickly, (and thanks to you Dave).

It's too bad Bob won't be in Las Vegas next week.

You didn't say, was he using Vista for the demo?

I suppose he got asked about the progress of the alpha testing.

Posted on Friday, May 25, 2007 by Russell Eggen

I just asked Bob that. He said he was not using Vista as there is some extra stuff he needs to address for ASP.NET on Vista.

Also, I neglected to point out in my write up that Bob actually showed code generation. Entirely my fault as I missed it (busy trying to keep my notes up <g>). I'll be posting a follow-up on this as Bob promised me that he will re-demo it to me.

Posted on Sunday, May 27, 2007 by Tony York

As the convenor of the Aussie Devcon, I just want to say that Bob Z's presentations far exceeded my expectations, and left the room speechless when he finished demonstrating what he could do with the new C7 and C7.Net. It really was something to behold.

Thanks Russ, for doing the reporting of the event for us.

Tony

Add a comment

Clarion Magazine

Calling Google Earth From Clarion

by Fernando Cerini

Published 2007-05-23

In this article I will demonstrate a very easy way to integrate Google Earth and Clarion. Basically you generate an XML file in a specific format (a KML file) and tell Google Earth, via an API call, to either open the file or refresh its display based on the file.

The ShellExecute API call

Your application communicates with Google Earth via the Windows ShellExecute API function. When you install Google Earth on your computer, the installation program adds a file association for filenames ending with .KML. When you call ShellExecute on a KML file, Windows passes the filename and the other specified information to Google Earth, launching the application if necessary. Each time you call ShellExecute, Google Earth gets a new file and will "fly" to the new coordinates.

You can also tell Google Earth to automatically read a specific file at regular intervals. You then update the file with new information as needed; this display mode is called "network link".

KML files

A KML file is an XML file containing a wide range of geographical information, from simple location data to 3D models and map markers. A basic KML file looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Folder>
  <name>Folder name</name>
  <description>Folder description</description>
  <Placemark>
    <name> Placemark Name</name>
    <description>Placemark Description</description>
    <Style>
      <IconStyle>
        <color>ffffffff</color>
```

```
                <scale>0.9</scale>
            </IconStyle>
            <LabelStyle>
                <color>ff00ffff</color>
                <scale>0.8</scale>
            </LabelStyle>
        </Style>
        <Point>
            <coordinates>lon,lat[,alt]</coordinates>
        </Point>
    </Placemark>
</Folder>
</kml>
```

You can create several Placemarks inside a Folder, and each Placemark can have different styles and coordinates.

Like any XML file, the KML file is basically a text file; you can use the ASCII driver to generate this file or you can use an XML library. If you don't mind getting your hands dirty, the fastest way to dynamically generate text files is using some API calls.

The code

Let's start with the code. First, you must declare the APIs in the Inside Global Map embed:

```
!Inside Global map
Module('Win32.lib')
ShellExecute(Long,*CString,*CString,*CString,*CString,Short),UShort,PASCAL,RAW,NAME
('ShellExecuteA')
_lcreat(*CSTRING,SIGNED),SIGNED,PASCAL,RAW
_hwrite(SIGNED,*CSTRING,LONG),LONG,PASCAL,RAW
_lclose(SIGNED),SIGNED,PASCAL
END
In our procedure, you must declare some local data
!Local Data
KML           CSTRING(999000)
KMLName       CSTRING(200)
!Shellexecute parameters
LOC:Handle LONG
```

```
LOC:Op     CSTRING (255)
LOC:File   CSTRING (255)
LOC:Path   CSTRING (255)
LOC:Param  CSTRING (255)
LOC:Show   LONG
LOC:RetHandle LONG
```

Here's a file definition for Global Positioning System (GPS) data storage:

```
GPSData             FILE,DRIVER('TOPSPEED'),PRE(RGPS),CREATE,BINDABLE,THREAD
Record               RECORD,PRE()
GPSDate                 DATE
GPSTime                 TIME
Info                 CSTRING(100)
Longitud               DECIMAL(9,6)
Latitud              DECIMAL(9,6)
                    END
                  END
```

Given the file definition above, here is some code to display a series of waypoints. You could call this code from a View in Map button:

```
CLEAR(RGPS:RECORD)
SET(GPSData)
Count#= 0
LOOP UNTIL ACCESS:GPSData.Next()
  Count# += 1
  IF Count#= 1
    !for more info about kml files:
    !http://earth.google.com/kml/
    KML = '<<?xml version="1.0" encoding="UTF-8"?><13,10>'&|
      '<<kml xmlns="http://earth.google.com/kml/2.1"><13,10>'&|
      '<<Folder><13,10>'&|
      ' <<name>Folder name<</name><13,10>'&|
      ' <<description>Folder description<</description><13,10>'
  END
  KML = KML&' <<Placemark><13,10>'&|
    ' <<name>'& FORMAT(RGPS:GPSDate, @D6)&|
    ' '& FORMAT(RGPS:GPSTime, @T1) &'<</name><13,10>'&|
    ' <<description>'& CLIP(RGPS:Info) &'<</description><13,10>'&|
    '<<Style><13,10>'&|
      '<<IconStyle><13,10>'&|
        '<<color>ffffffff<</color><13,10>'&|
        '<<scale>0.9<</scale><13,10>'&|
      '<</IconStyle><13,10>'&|
      '<<LabelStyle><13,10>'&|
        '<<color>ff00ffff<</color><13,10>'&|
        '<<scale>0.8<</scale><13,10>'&|
      '<</LabelStyle><13,10>'&|
    '<</Style><13,10>'&|
    '  <<Point><13,10>'&|
    '   <<coordinates>'& RGPS:Longitud &','& RGPS:Latitud &|
    ',0<</coordinates><13,10>'&|
    '  <</Point><13,10>'&|
    ' <</Placemark><13,10>'
END
IF Count# THEN
  KML = KML&'<</Folder><13,10><</kml><13,10>'
  KMLName =  LONGPATH() &'\GPSData.KML'
  F# = _lcreat(KMLName,0)
  X# = _hwrite(F#,KML,LEN(KML))
  X# = _lclose(F#)

  LOC:Handle = 0{PROP:Handle}
  LOC:Op     = 'Open'
  LOC:File   = KMLName
  LOC:Path   = LONGPATH()
  LOC:Param  = ' '
  LOC:Show   = 1
  LOC:RetHandle =  ShellExecute(LOC:Handle,LOC:Op,LOC:File,|
                     LOC:Param,LOC:Path,LOC:Show)
END
```

Although the XML code looks complex, it's not really that difficult to code. The file data is shown in bold, and at a minimum that's all you need to change to make this code work with your own database.

Each time you press the button, Google Earth will "fly" to a view where you can see all of the plotted points (Figure 1).

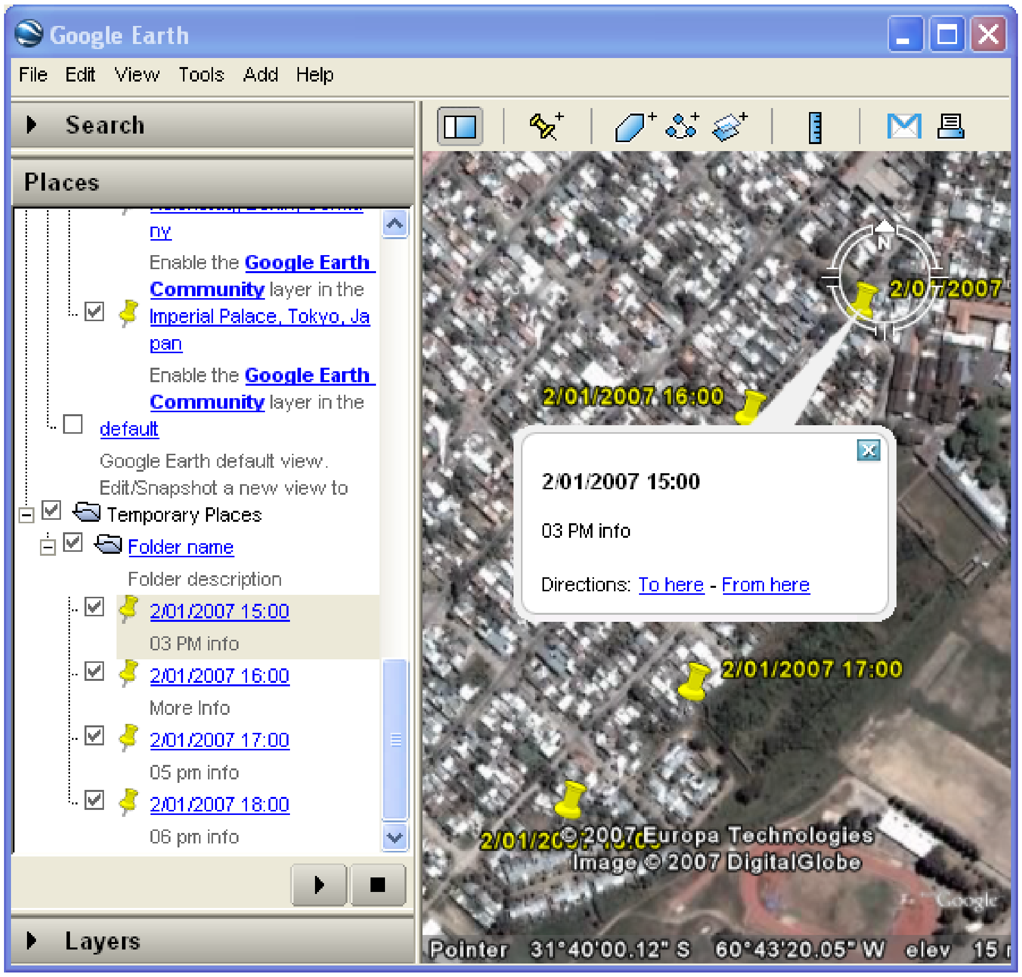

Google Earth
File Edit View Tools Add Help
Search
Places
Enable the Google Earth Community layer in the
Imperial Palace, Tokyo, Japan
Enable the Google Earth Community layer in the
default
Google Earth default view. Edit/Snapshot a new view to
Temporary Places
Folder name
Folder description
2/01/2007 15:00
03 PM info
2/01/2007 16:00
More Info
2/01/2007 17:00
05 pm info
2/01/2007 18:00
06 pm info
Layers
2/01/2007 16:00
2/01/2007 15:00
03 PM info
Directions: To here - From here
2/01/2007 17:00
©2007 Europa Technologies
Image © 2007 DigitalGlobe
Pointer 31°40'00.12" S 60°43'20.05" W elev 15

Figure 1. Displaying plotted points in Google Earth

You can use this technique to show the route of some vehicle in the map, or to make Google Earth fly and zoom to show the last position of one vehicle.

Using a Network Link

The other way to trigger this display it is to generate the KML file at regular intervals using a timer, and tell Google to refresh the information with a Network Link pointing to this file.

Right-click on a folder in the My Places Panel. Select Add > Network Link from the pop-up menu (Figure 2). When you add a network link in this way, the selected folder is automatically set as the container for the network link.

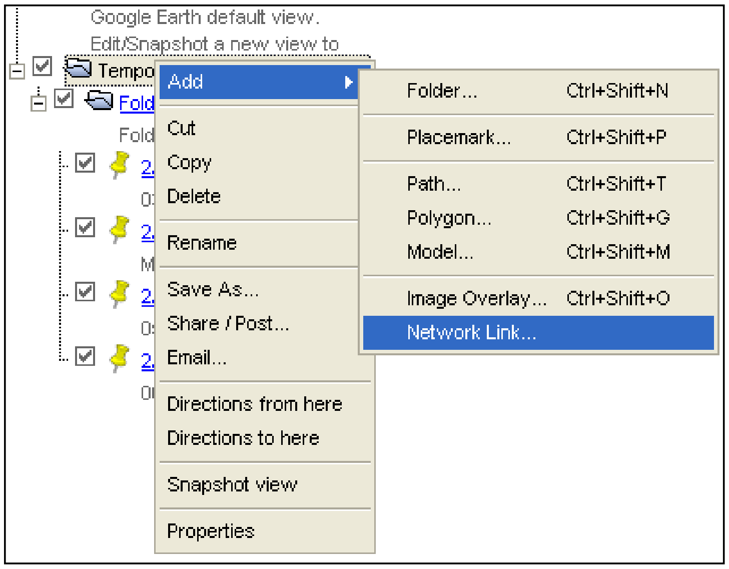
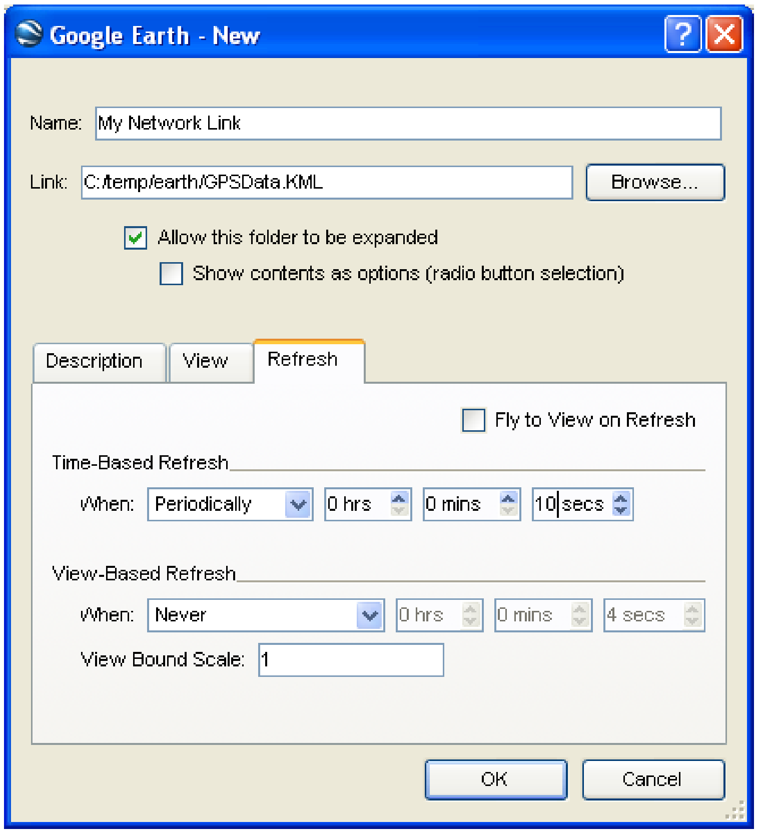


Figure 2. Adding a network link

The New Network Link dialog box appears (Figure 3). Enter the name of your link in the Name field. Enter the full path of the KML file in the Link field, or browse to the file location if the file is located on a network.


Google Earth - New

Name: My Network Link

Link: C:/temp/earth/GPSData.KML

[✓] Allow this folder to be expanded
[ ] Show contents as options (radio button selection)

Description | View | Refresh

[ ] Fly to View on Refresh

Time-Based Refresh
When: Periodically 0 hrs 0 mins 10 secs

View-Based Refresh
When: Never 0 hrs 0 mins 4 secs
View Bound Scale: 1

Figure 3. Network link properties

For example, if you collect the information from the vehicles' GPSs, you can generate the KML with the last position of each vehicle. And with the Network Link Google Earth will refresh this information automatically (see Figure 4).

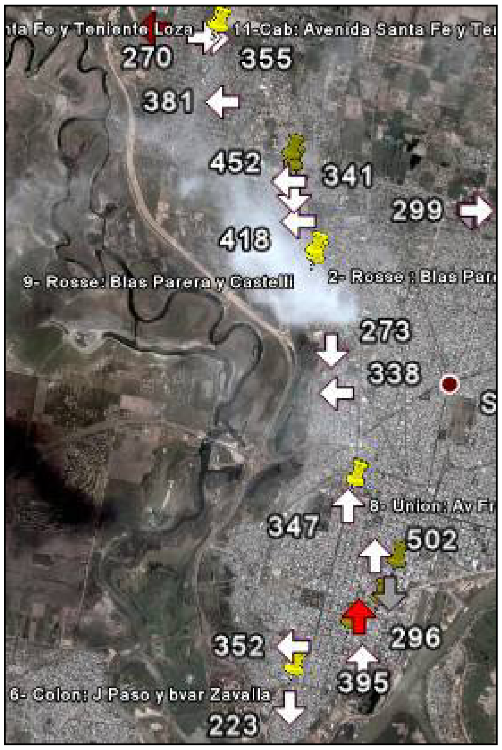


Figure 4. GPS data plotted on Google Earth via a network link

Summary

Your Clarion applications can easily control Google Earth's display by calling the Windows ShellExecute API function, and Google Earth's XML format files are fairly easy to create even if you don't know XML.

As I said, this is just a Getting Started article. I hope you found this information useful, and don't hesitate to contact me if you have any questions.

Download the source

Fernando Cerini is a Software Engineer with more than 10 years of experience in Clarion. Until 2003, he was in charge of research, support and training for the local Clarion distributor in Argentina. Now he works for Evolution Consulting.com, the Clarion International Distributor for Latin America, which has carried out numerous courses and conferences in the region.

Reader Comments

Posted on Saturday, May 26, 2007 by David Groves

Very interesting article Fernando.
Is there a way to get the GPS of the place you are looking at in Google and store within your own database ?

The process would be something like
1. Fly to an address / location
2 Get the GPS for that address
3. Store GPS in database

Posted on Sunday, May 27, 2007 by Fernando Cerini

Well, if you have several Placemarks, you can save the Placemarks as a kml file and then process the kml file in clarion, with the ASCII Driver.

If you need to enter one placemark, you can copy the placemark, and then parse the clipboard from clarion.

Posted on Monday, May 28, 2007 by Mike McLoughlin

Do you need to subscribe to the Pro version of Google Earth to be able to interface like this? Or will the free version do?

Posted on Monday, May 28, 2007 by Fernando Cerini

Hi Mike,
This article is based on the free version.

Add a comment

Clarion Magazine

Sending A String Between Processes

by Larry Sand

Published 2007-05-22

In the previous articles in this series I described the implementation of the InterprocessComs class with its ability to send string data between processes. Now I'll show you how to use it in your programs, starting with sending a string between processes.

Here's the program responsible for sending the string:

```
  Program
  Include('ipcCl.inc'),Once

  Map
  End
W    WINDOW('Interprocess communication, Send string data')|
       ,AT(,,300,200),SYSTEM,GRAY
     BUTTON('Send Data'),AT(30,33,83,14),USE(?SendData)
   END
LINK_GUID   Equate('F169F831-8447-4b89-BCEF-54FCDF09E83D')
UM_GUID     Equate('DC9DE21E-5A75-4991-B25E-691FB98BF26B')

ipc InterprocessComs
s   String(128)
  Code

  Open(W)
  s = 'Clarion Magazine rocks!'
  If ipc.Init(W, LINK_GUID)
    ipc.RegisterUserMessage(UM_GUID)
  End
  Accept
    Case Field()
    Of ?SendData
      If EVENT() = EVENT:Accepted
        If ipc.SendData(s, Size(s))
          Message('String successfully sent to partner',|
         'Ipc send string')
        End
      End
    End

  End
  Close(W)
  Return
```

The basic use is the same as discussed before. Ipc is an instance of the InterprocessComs class. You declare a String s, initialize the ipc object, and register a message. The user message isn't required in this example and you can omit it if you wish. In the button's accepted event the SendData method is called passing the string and the size of the string (Size(s)). You can also send the length of the string and less data are copied. To do that, substitute Len(Clip(s)) for Size(s) as the cbData parameter. In this case you'll only copy 23 bytes as opposed to 128 bytes if you used Size(s).

That's how you send the string. Here's how you receive it in a very simple program:

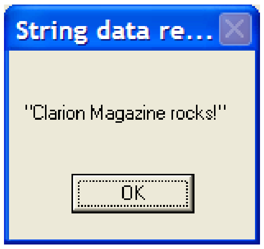
```
  Program
  Include('ipcCl.inc'),Once

  Map
  End
W    WINDOW('Interprocess communication, receive string data')|
       ,AT(,,300,200),SYSTEM,GRAY
     END
LINK_GUID   Equate('F169F831-8447-4b89-BCEF-54FCDF09E83D')
UM_GUID     Equate('DC9DE21E-5A75-4991-B25E-691FB98BF26B')

ipc InterprocessComs
s   String(128)
  Code
  Open(W)
  If ipc.Init(W, LINK_GUID)
    ipc.RegisterUserMessage(UM_GUID)
    ipc.SetReceiveBuffer(s)
  End
  Accept
   Case EVENT()
   Of WMU_IPC_STRINGRECEIVED
     message('"'& Clip(s) &'"','String data received')
   End
  End
```

```
    Close(W)
    Return
```

All of the declarations and initialization of the object are the same as the other examples so I won't discuss them here. Note that a string s is declared identically to the one in the sending program. After the object is initialized the SetReceiveBuffer method is called passing s so it can save the reference to it in the sData property. When the WM_COPYDATA message is processed by the ProcessCopyDataStruct method, ProcessCopyDataStruct sends the WMU_IPC_STRINGRECEIVED message to the ACCEPT loop where you can catch it with EVENT(). This program displays the string a message box, as in Figure 1.



"Clarion Magazine rocks!"

Figure 1. Message received

However, there's a glaring error with this code – did you spot it?

While this message box is displaying the received string your program will continue to receive WM_COPYDATA messages in the windows procedure, and ProcessCopyDataStruct will copy the data to the string s. It will also POST WMU_IPC_STRINGRECEIVED messages to your ACCEPT loop and each of them will be queued up and received. However, only the *last* string copied into s is available. Clearly this isn't what you want to happen.

Your program needs to have a way to store these data as they are received. Then allow you to access to each datum when it's convenient. To accomplish this the InterprocessComs class needs someplace to temporarily store the received data. Then you'll need a method to retrieve the data as the WMU_IPC_STRINGRECEIVED messages are processed by your ACCEPT loop. You'll also need to treat the data store as unthreaded global data; because two processes may access it simultaneously, you'll need to synchronize access.

Here's how to implement the data store as a FIFO (First In First Out) queue whose access is protected by a critical section and send the data in signed packets.

To begin with you'll need a queue to store the received strings before they're processed in your receiving programs' ACCEPT loop. The queue only needs to store a string of unknown length. To do this, declare a named queue with a String reference variable:

```
ipcDataQueue Queue,Type
sData        &String
             End
```

The reference variable is the only element necessary to implement a FIFO queue.

The class needs two new properties, one for the above queue and another for the critical section used to synchronize access. They are declared like this:

```
DataQ      &ipcDataQueue,Protected
ipcDataQcs &ICriticalSection,Protected
```

The critical section interface is declared in the CwSynchM.inc file that's located in your LibSrc folder. It's included in the ipcCl.inc class header file with this code:

```
Include('CwSynchM.inc'),Once
```

When the object is instantiated, it's constructor is called before any other methods and you'll use it to create the queue and critical section. This code creates them:

```
Self.DataQ &= New ipcDataQueue
Self.ipcDataQcs &= NewCriticalSection()
```

There's some additional error handling code to the Init method to check that these reference variables are not null. The code's self explanatory and not described here. Whenever you allocate memory in a constructor, it's a good habit to write the cleanup code before you forget. Here's the new destructor that takes care of freeing the resources.

```
InterprocessComs.Destruct          Procedure()
i   Long,Auto
  Code
  If Not Self.DataQ &= Null
    Loop i = 1 To Records(Self.DataQ)
      Get(Self.DataQ, i)
      Dispose(Self.DataQ.sData)
    End
    Dispose(Self.DataQ)
  End

  If Not Self.ipcDataQcs &= Null
    Self.ipcDataQcs.Kill()
  End

  Return
```

If the DataQ still contains records you must free the memory pointed to by its sData string reference variable. After all the strings are freed, the queue pointed to by DataQ is disposed. If the critical section was created you call its Kill method to free its resources. See the Clarion documentation for information on the critical section interface.

The change to storing received data in a FIFO queue requires a complete overhaul of the ProcessCopyDataStruct method Here's the code for the revised method:

```
InterprocessComs.ProcessCopyDataStruct     |
  Procedure(CMAG_COPYDATASTRUCT cds)!,UNSIGNED,Virtual
```

```
Result  UNSIGNED,Auto
 Code
 Result = False
 Case cds.dwData
 Of WMU_IPC_STRINGDATA
  If Not Self.ipcDataQcs &=Null And Not Self.DataQ &= Null
   Self.ipcDataQcs.Wait()
    Clear(Self.DataQ)
    Self.DataQ.sData &= New String(cds.cbData)
    If Not Self.DataQ.sData &= Null
     CMAG_MemCpyRD(Self.DataQ.sData, cds.lpData, cds.cbData)
     Add(Self.DataQ)
     If Not Errorcode()
      Result = True
      Post(WMU_IPC_STRINGRECEIVED)
     Else
      Self.TakeError(IPC_ERROR_OUTOFMEMORY, 'Add data queue')
     End
    Else
     Self.TakeError(IPC_ERROR_OUTOFMEMORY, 'Allocate data string')
    End
   Self.ipcDataQcs.Release()
  End
 End
 Return Result
```

In case the object wasn't properly initialized you test to see that the queue and critical section reference properties are not null. If they're valid, you wait for the critical section. You'll only wait if another method has not released the critical section. Once you have control of the critical section it's safe to access the queue buffer. First it's cleared and then you allocate a string that is cds.cbData bytes in size. At this point if Self.DataQ.sData is not null you have a valid buffer into which you may copy the memory from cds.lpData using MemCpy as described before. Then the DataQ record is saved and the WMU_IPC_STRINGRECEIVED message is posted to the ACCEPT loop. When done the critical section is released by calling its Release method. It's important to keep the code between Wait and Release as small as possible to minimize contention for the critical section.

That's how the received strings are added to the end of the DataQ. Now I'll show you how to pop them off the top to complete the FIFO implementation. This is done in the new GetData method.

```
InterprocessComs.GetData   Procedure(*String sData)!,Virtual
 Code
 sData = ''
 If Not Self.DataQ &= Null And Not Self.ipcDataQcs &=Null
```

```
      Self.ipcDataQcs.Wait()
        If Records(Self.DataQ)
          Get(Self.DataQ, 1)
          If Not Self.DataQ.sData &= Null
            sData = Self.DataQ.sData
            If Size(sData) < Size(Self.DataQ.sData)
              Self.TakeError(IPC_ERROR_DATATRUNCATED,'GetData')
            End
            Dispose(Self.DataQ.sData)
          End
          Delete(Self.DataQ)
        End
      Self.ipcDataQcs.Release()
    End
    Return
```

If the DataQ and ipcDataQcs reference variables are not null, wait on the critical section. After you have control of the critical section (Wait returns), Get the first record in the DataQ and check that its sData string reference variable is not null. If everything is okay assign the queue's string to the sData string that was passed as a parameter to the method. If the destination variable is smaller than the source, the method throws an error. Then the memory allocated for the string in the queue is released by calling Dispose. Finally the queue record is deleted and the critical section is released.

That's all there is to creating a FIFO queue; you always read record number one and as soon as you're done with it you delete the record. All new records are added to the end of the queue and you "pop" them off as they're read.

The SetReceiveBuffer method was removed as it's no longer needed

Even though you can send a string without any other checking I suggest that you create a packet of data using a group that contains a signature and the type of data included in the group. This creates a header that describes the data in the message. Consider this group definition:

```
ipcPacketHeader Group,Type
guid            String(36)
dataType        Long
                End
```

You can probably guess from the declaration that the signature is a GUID. Since you already have at least one GUID that's shared between the two processes, you'll use it to sign the message and then you'll check that signature when it's received. You can be certain that you sent the message if the two GUIDs match.

To use the header you'll declare a group derived from it in both programs. To keep the declarations in sync you'll use a separate header file that you can include in both programs. The example program declares a packet group like this:

```
myPacket  Group(ipcPacketHeader)
```

```
sData       String(128)
         End
```

myPacket inherits the fields in ipcPacketHeader and you have access to the three of them using dot notation (e.g. myPacket.guid). Given the myPacket declaration above, consider this group:

```
ipcOrderInfo Group,Type
OrderDate       Long
OrderTime       Long
OrderNumber     Long
            End
myOrderInfo     Like(ipcOrderInfo),Over(myPacket.sData)
```

myOrderInfo redefines the first 12 bytes of myPacket.sData as a group of three long integers. You're not limited to this definition, you can declare any group where Size(myNewGroup) <= Size(myPacket.sData).

The example program will send three types of data in the packet's string sData. To differentiate between them in the receiving program, you'll declare some constants to pass in myPacket.dataType. The three data types are:

```
   Itemize
IPC_PACKET_DATA_CUSTNAME    Equate
IPC_PACKET_DATA_PHONENUMBER Equate
IPC_PACKET_DATA_ORDERINFO   Equate
   End
```

You can define your own packet and data types in your own applications. That is, the data are completely user defined. They are not defined in the class. The ipcPacketHeader type declaration is in the class header file, however, you don't have to use it. Feel free to define your own.

In the source zip you'll find two example projects, ipcCDS.prj (Copy Data Send) and ipcCDR.prj (Copy Data Receive). They are very similar to the above code that had the synchronization error. You'll find the code in ipcCDR.clw, ipcCDS.clw, ipcPkt.inc, ipcCl.inc, and ipcCl.clw.

The sending program ipcCDS allows you to choose one of the three packets defined earlier and choose how many of them to send at one time. If you set this to a very large number you'll see that it is possible to overload the message queue of the receiver's ACCEPT loop. Window's allocates resources for 10,000 messages in the queue. This limit is for all of the messages it's processing not only the ones you're sending. Posted messages that exceed this limit are discarded, however the data are still written to the FIFO receive queue. If you think this may be a limitation for your program then you'll need to modify the code to retrieve the message data from the object's receive buffer.

To send a myPacket of data you'll fill in the elements of the group, for example, this is how the information is assigned to the IPC_PACKET_DATA_CUSTNAME type packet:

```
myPacket.guid = LINK_GUID
myPacket.dataType = IPC_PACKET_DATA_CUSTNAME
myPacket.sData = 'Clarion Magazine'
```

Then call the SendData method:

```
ipc.SendData(myPacket, Size(myPacket) )
```

The receiving program ipcCDR, displays the messages, as they are received, in a list box. Remember from the earlier discussion of receiving WM_COPYDATA, that you must process the WMU_IPC_STRINGRECEIVED event (message) that the InterprocessComs class posted to your ACCEPT loop to remove the data from the object's FIFO buffer. Here's the code that tests for the message in the ACCEPT loop and copies them into the MessageQ queue that the listbox displays:

```
Case EVENT()
Of WMU_IPC_STRINGRECEIVED
  ipc.GetData(myPacket)
  If myPacket.guid = LINK_GUID
    Case myPacket.dataType
    Of IPC_PACKET_DATA_CUSTNAME
       MessageQ.s = 'Name: ' & myPacket.sData

    Of IPC_PACKET_DATA_PHONENUMBER
     MessageQ.s = 'Phone: ' & myPacket.sData

    Of IPC_PACKET_DATA_ORDERINFO
      MessageQ.s = 'Date: '& Format(myOrderInfo.OrderDate,@D17) |
         &' Time: '& Format(myOrderInfo.OrderTime,@T4) |
         &' Number: '& myOrderInfo.OrderNumber
    End

    Add(messageQ)
  End
End
```

When an WMU_IPC_STRINGRECEIVED event (message) is received you call the GetData method, passing your myPacket group. It copies the received data into the group and removes it from the object's FIFO queue . After the data is in your group you can check the GUID (here I've used the LINK_GUID that's defined in ipcPkt.inc). When the GUIDs match you can check the dataType of the packet and access these data according to how they were defined. For the two string data types, IPC_PACKET_DATA_CUSTNAME and IPC_PACKET_DATA_PHONENUMBER, you can access the sData element of the myPacket group directly. On the other hand, IPC_PACKET_DATA_ORDERINFO data type is defined as a group of three longs (the myOrderInfo group). To retrieve these data as longs you'll need to use myOrderInfo.OrderDate, myOrderInfo.OrderTime, and myOrderInfo.OrderNumber.

Compile both programs and start them. Press the Send Data button and you'll see the messages appear almost instantly in the receiving program's listbox.

To implement my final version of the class to pass messages in your program follow these steps:

1. Add Include('ipcCl.inc'),Once globally
2. Instantiate the class in the data section of your top level window using any label you wish. Here it is declared as:
   ipc InterprocessComs
3. Declare two GUID constants (use guidgen.exe to create your own GUIDs):
   LINK_GUID Equate('F169F831-8447-4b89-BCEF-54FCDF09E83D')
   UM_GUID Equate('DC9DE21E-5A75-4991-B25E-691FB98BF26B')
4. Define your data. This can be anything you need to send to your partner process.
5. After the window is opened call the Init method, passing it the label of the window and your GUID:
   ipc.Init(MyWindow, LINK_GUID)
6. If the ipc object was successfully initialized, and you're going to use a user message, register it by passing the ipc object another GUID with the RegisterUserMessage method:
   ipc.RegisterUserMessage(UM_GUID)
7. In the sending program, assign your data to the group and call the SendData method to transmit the group to your receiving program:
   ipc.SendData(myPacket, Size(myPacket))
8. In the receiving program, catch the message notifying you that there's a message in the FIFO receive buffer by using Clarion's EVENT function in the ACCEPT loop:
   Case EVENT()
   Of WMU_IPC_STRINGRECEIVED
9. After you receive WMU_IPC_STRINGRECEIVED, call the object's GetData method:
   ipc.GetData(myPacket)
10. Do something with your data as soon as it's received. When the ACCEPT loop cycles you may receive another WMU_IPC_STRINGRECEIVED event (message). Depending on your design you'll either act on it and/or make a copy of it to use later.

In this series of articles you've learned about windows messages, including how they're sent between processes using the asynchronous PostMessage function, and the synchronous SendMessage and SendMessageTimeout functions. You also learned how to use POST and NOTIFY/NOTIFICATION for interthread messages. Next, you learned how you to receive messages. Interprocess messages required that you subclass the window procedure and you learned to do that in a thread safe manner using a procedure in a class member module. This required that you set and get window properties using Windows API functions; SetProp, GetProp and then when done RemoveProp.

After you were comfortable sending and receiving messages, you saw how they could be used to asynchronously communicate state changes between processes, including sending a single 32 bit value along with the message. Then you learned how to send more than a single 32 bit value using WM_COPYDATA and the synchronous SendMessageTimeout function. The WM_COPYDATA message allows you to send any data between processes. Because it's possible to receive this message while you're accessing the data you learned how to protect it with a critical section and implement a FIFO queue to store the received messages while you're processing their data.

Hopefully you have enough information to extend the class and framework that I presented for use in your programs. I think that you'll find the class quite flexible.

Download the source

Larry Sand is an independent software developer who began programming with Clarion in 1987. In addition to normal database development, he specializes in connecting Clarion to external devices like SCUBA diving computers, kilns, and satellite transceivers used in medical helicopters. In other lives, he sailed Lake Superior as the owner/operator of shipwreck SCUBA diving tours and later as a Master for the Vista Fleet. When Larry is not programming you'll find him messing about in boats, or with boats.

Reader Comments

Add a comment

Clarion Magazine

Coping With Vista's "Virtual" Reality, Part 2

by Jane Fleming

Published 2007-05-14

In the previous installment in this series I looked at the virtualization of files and folders. But that's just part of the story in Vista's virtual world. Microsoft's graphic (Figure 10) illustrates that virtualization affects the registry similarly.

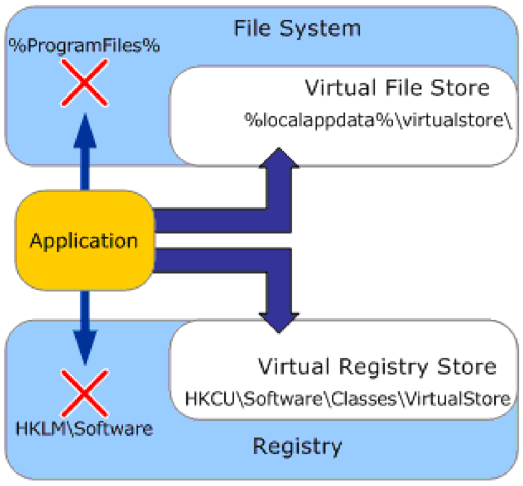


Figure 10. Virtualization affects both file system and registry

In the pre-Vista world, using an administrator account was all you needed to do to write to the parts of the registry that are not individualized, such as HKEY_LOCAL_MACHINE. Vista requires elevation. But again, if you don't have a Vista-aware manifest in your app, Vista will bring out the smoke and mirrors.

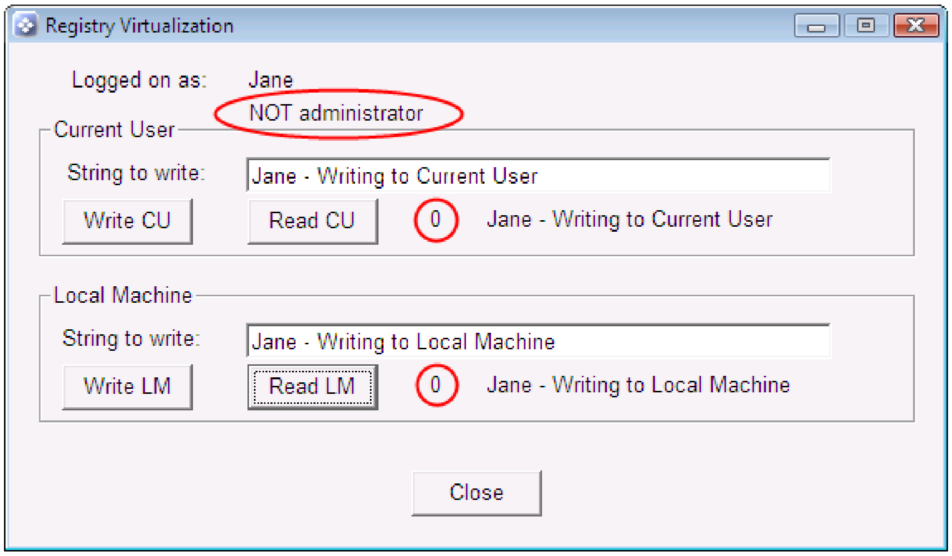
Registry Virtualization Demo

Compile vtest.prj from the downloadable source zip. This small application lets you try to write to and read from both the CURRENT_USER and LOCAL_MACHINE sections of the registry. (If you're going to run it on a Windows 2000 machine, comment out the code as indicated in vtest.clw. Uncomment those lines and recompile before running on Vista.)

The String to Write fields are your input. The strings beneath them (to the right of the return value numbers) are what's read from those registry keys. The Write CU and Write LM buttons attempt to write your string to the registry using Clarion's PutReg function then read it back with GetReg and display it to the right of the return value. The PutReg function returns 0 if it succeeds, a non-zero value otherwise.

The program is hard-coded to write one key value in HKLM and one in HKCU, each labeled Software \Beach Bunny Software\TestStuff. Feel free to create a differently-named key if you prefer.

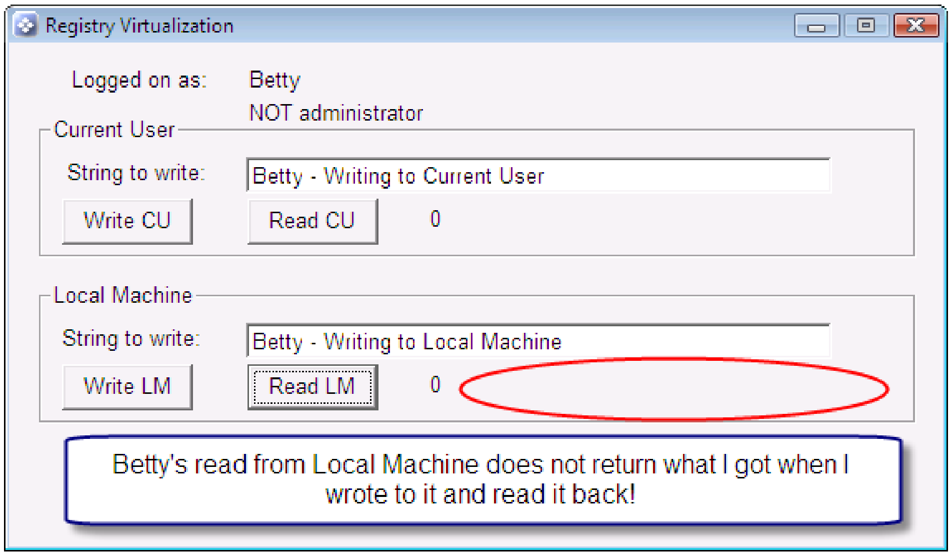
In Figure 11, I'm running the program on a Vista machine. The program does not have a Vista-aware manifest. While my account is a member of the Administrators group, the call to IsUserAnAdmin() correctly reports that I'm not functioning as an administrator. However, both of the strings I've written (to CURRENT_USER and to LOCAL_MACHINE) show as successful – the return value is zero and the strings are read back correctly.



Jane
NOT administrator
Jane - Writing to Current User
0 Jane - Writing to Current User
Jane - Writing to Local Machine
0 Jane - Writing to Local Machine

Figure 11. Although I'm not an administrator, Vista shows good return values and reads back the data

If I've really written to LOCAL_MACHINE**,** Betty should see the value I wrote ("Jane – Writing to Local Machine") when she clicks Read LM under her logon. But again, we've got smoke and mirrors.

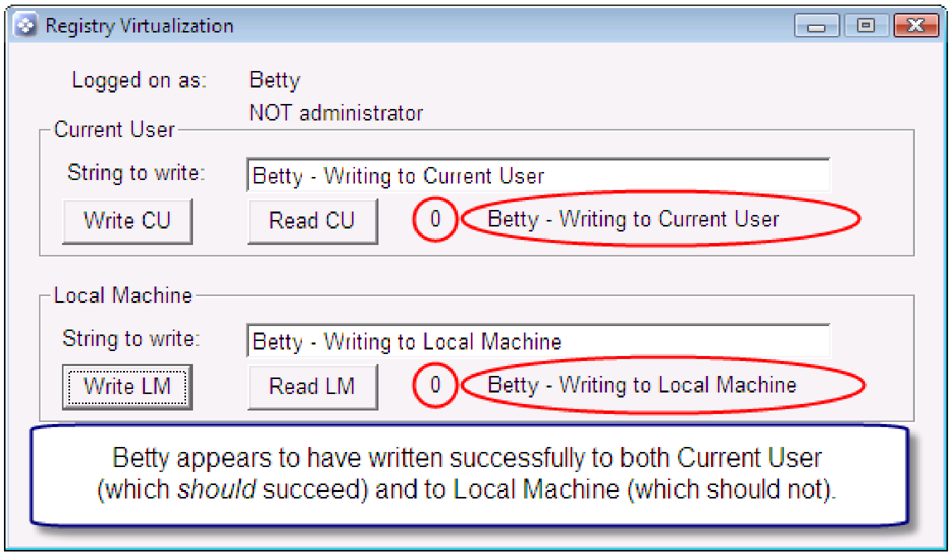


Betty
NOT administrator
Betty - Writing to Current User
0
Betty - Writing to Local Machine
0

Betty's read from Local Machine does not return what I got when I wrote to it and read it back!

Figure 12. Another user cannot read the previously written Local Machine entry that's been virtualized

Now it's Betty's turn. She also seems to have written to Current User and, without being a member of the administrators group, to Local Machine (Figure 13).



Logged on as: Betty
NOT administrator

Current User
String to write: Betty - Writing to Current User

0 Betty - Writing to Current User

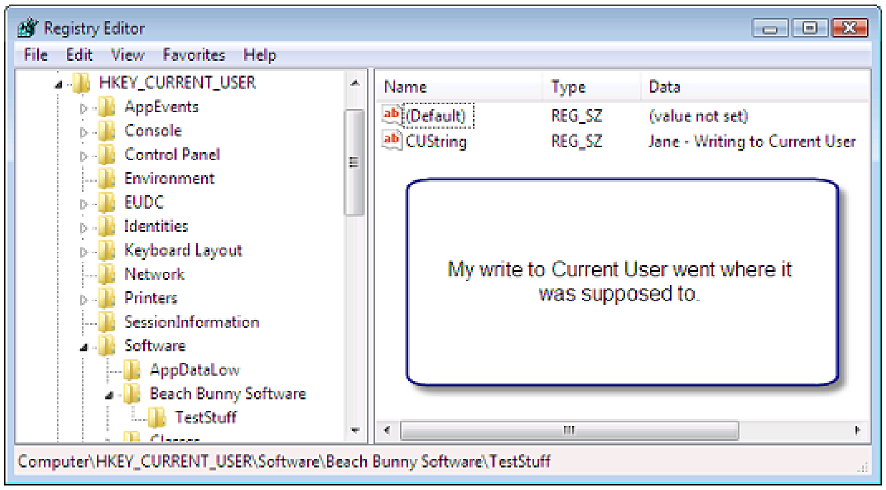
Local Machine
String to write: Betty - Writing to Local Machine

0 Betty - Writing to Local Machine

Betty appears to have written successfully to both Current User (which *should* succeed) and to Local Machine (which should not).

Figure 13. The appearance of success

But what's actually in the registry? In Figure 14 I've logged on as Jane, and I can see the Current User record is as expected.



HKEY_CURRENT_USER
AppEvents
Console
Control Panel
Environment
EUDC
Identities
Keyboard Layout
Network
Printers
SessionInformation
Software
AppDataLow
Beach Bunny Software
TestStuff

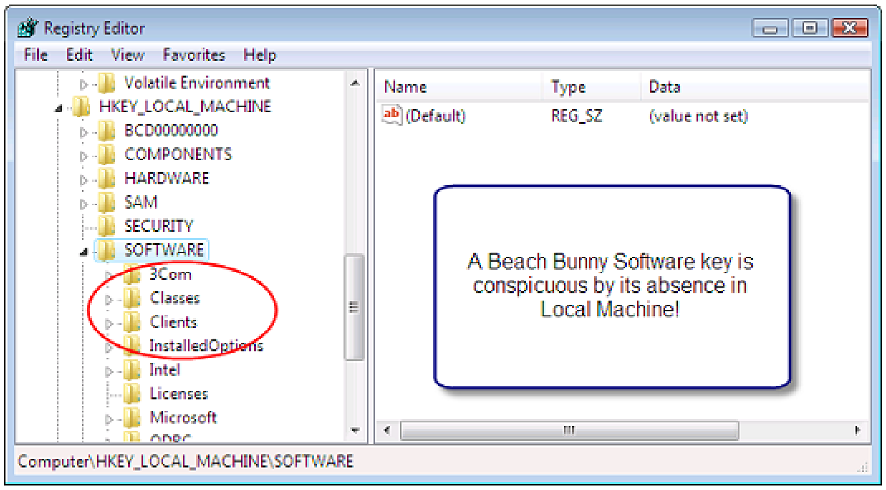
Name	Type	Data
(Default)	REG_SZ	(value not set)
CUString	REG_SZ	Jane - Writing to Current User

My write to Current User went where it was supposed to.

Computer\HKEY_CURRENT_USER\Software\Beach Bunny Software\TestStuff

Figure 14. Logged on as Jane, current user key is correct (view full size image)

But as it turns out, neither Betty nor I actually wrote to the Local Machine key (Figure 15).



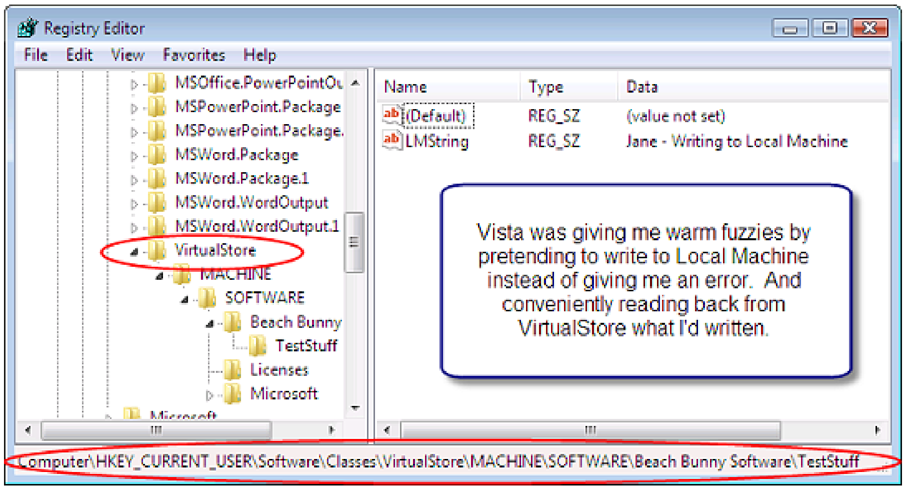
(Default) REG_SZ (value not set)

A Beach Bunny Software key is conspicuous by its absence in Local Machine!

Computer\HKEY_LOCAL_MACHINE\SOFTWARE

Figure 15. Proof that neither Betty nor I wrote to Local Machine (view full size image)

Virtual reality strikes again! My virtualized write is in the corresponding key in my Current User hive.



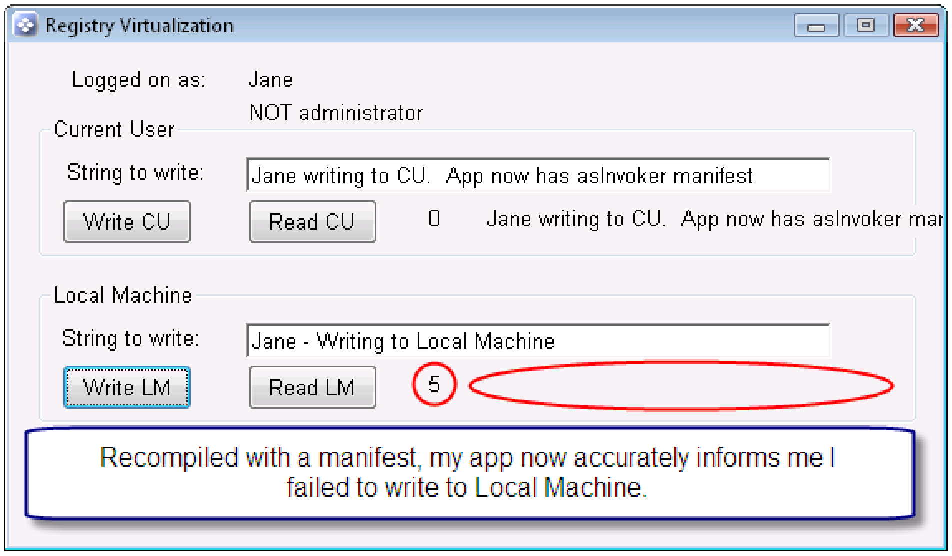
(Default) REG_SZ (value not set)
LMString REG_SZ Jane - Writing to Local Machine

Vista was giving me warm fuzzies by pretending to write to Local Machine instead of giving me an error. And conveniently reading back from VirtualStore what I'd written.

Computer\HKEY_CURRENT_USER\Software\Classes\VirtualStore\MACHINE\SOFTWARE\Beach Bunny Software\TestStuff

Figure 16. The Local Machine key is virtualized underVirtualStore (view full size image)

If, however, I use a Vista-aware manifest with my application, Windows informs my app that it failed to write to the Local Machine key (Figure 17).



Logged on as: Jane
NOT administrator

String to write: Jane writing to CU. App now has asInvoker manifest

0 Jane writing to CU. App now has asInvoker mar

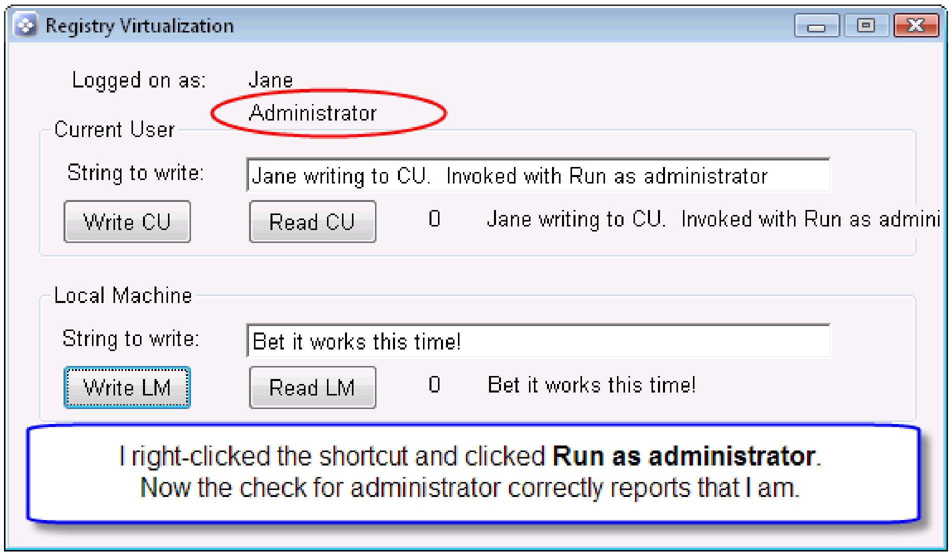
String to write: Jane - Writing to Local Machine

5

Recompiled with a manifest, my app now accurately informs me I failed to write to Local Machine.

Figure 17. App with manifest gets PutReg error

In Figure 18 I've run the app as administrator, and the PutReg to the Local Machine key works.



Logged on as: Jane
Administrator

String to write: Jane writing to CU. Invoked with Run as administrator

0 Jane writing to CU. Invoked with Run as admini

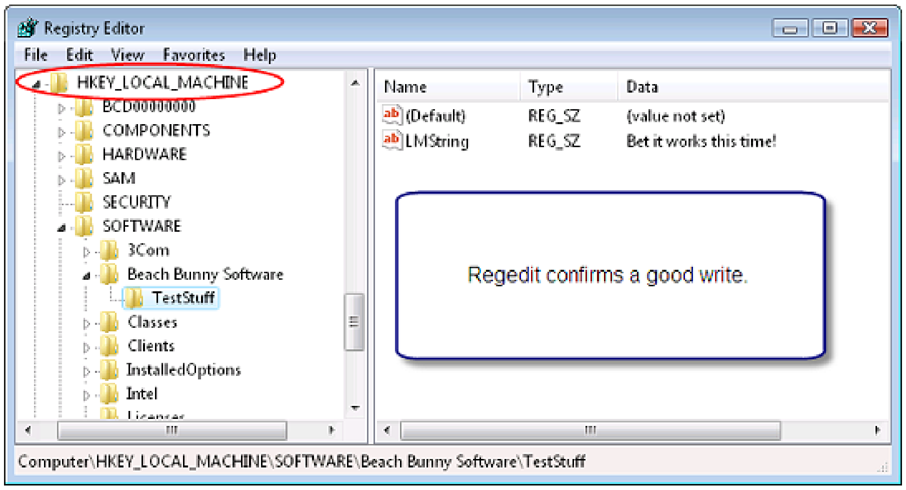
String to write: Bet it works this time!

0 Bet it works this time!

I right-clicked the shortcut and clicked **Run as administrator**. Now the check for administrator correctly reports that I am.

Figure 18. With the app running elevated, the PutReg return values are accurate

Figure 19 shows that the elevated application did in fact write to the Local Machine key.



Name	Type	Data
(Default)	REG_SZ	(value not set)
LMString	REG_SZ	Bet it works this time!

Regedit confirms a good write.

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Beach Bunny Software\TestStuff

Figure 19. Regedit confirms Local Machine key (view full size image)

Detecting Vista

One more thing I should touch on briefly is how apps can tell whether they're running under Vista. The GetVersionExA API call gives you the information you need. A simple example is included in IsVista.prj.

Figure 20 shows the test procedure running on a Vista machine.

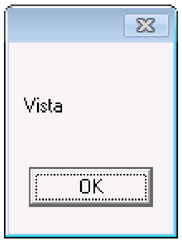
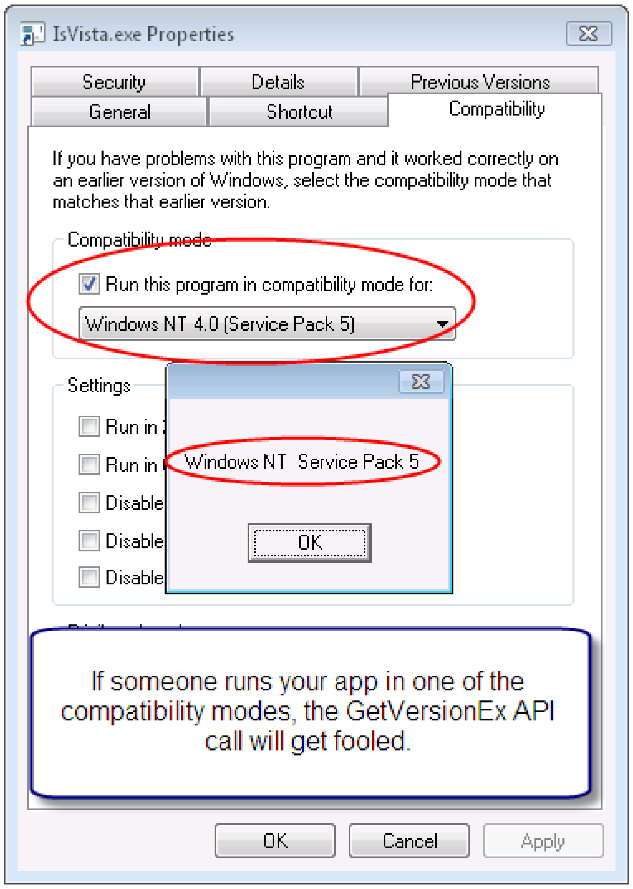

Vista

Figure 20. Sometimes a cigar is a cigar

But it's not that simple. If somebody has set one of the compatibility modes for the application, Vista will report that OS to the app. (By the way, the current compatibility mode may not be visible if it's been set on the All Users window rather than on the main Compatibility window. See Figure 8 in There's A Manifest In Your Destiny for a look at the All Users window.)


IsVista.exe Properties

If you have problems with this program and it worked correctly on an earlier version of Windows, select the compatibility mode that matches that earlier version.

Compatibility mode

☑ Run this program in compatibility mode for:

Windows NT 4.0 (Service Pack 5)

Settings

☐ Run in
☐ Run in
☐ Disable
☐ Disable
☐ Disable

Windows NT Service Pack 5

If someone runs your app in one of the compatibility modes, the GetVersionEx API call will get fooled.

Figure 21. Sometimes a cigar isn't a cigar

One Caveat

Microsoft regards virtualization as a temporary palliative. It is not guaranteed to be present in future versions of Windows. It is not used in 64-bit Vista now.

Lest We Forget Armadillo…

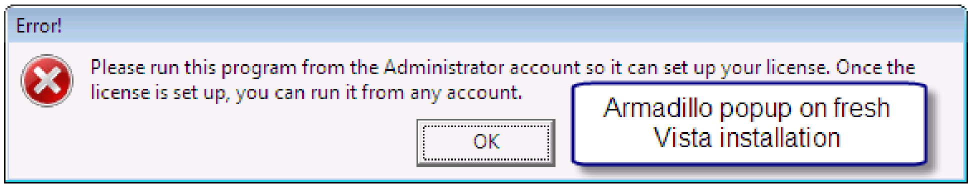
If you're using Armadillo to protect your applications you need to be aware of some additional virtualization issues. It's no secret that by default, Armadillo stores some key information in the common parts of the registry. There is an option on the Backup Key Options tab using the classic Armadillo interface called Use USER keys instead of SYSTEM, which would write to Current_User rather than to the registry hives that are common to all users. This is not Armadillo's default, because if your software offers a default certificate with a free trial period, all that's needed to get around a license stored in Current_User is to create a new user account.

Armadillo's specific mechanisms are somewhat secret. Some users on the Armadillo forums observed that on a clean installation of Vista (‘clean' meaning no Armadillo apps have ever been installed on the box), even *without* ticking the USER checkbox an application that *does not have a Vista-aware manifest* will exhibit this same behavior of giving you a new free trial each time you create a new user account.

By now, you should be able to figure out why. Vista is virtualizing Armadillo's attempts to write to the non-individual areas of the registry. So each new user account appears to Armadillo like a fresh registry.

So, how to deal with this?

You can embed a Vista-aware manifest in your app. That way, Vista won't virtualize and Armadillo will insist that your program be run the first time using the elevated Administrator token (Figure 22).


Error!

Please run this program from the Administrator account so it can set up your license. Once the license is set up, you can run it from any account.

Figure 22. Armadillo-protected app that has a Vista-aware manifest, running on a 'clean' Vista machine

Once you've run one Armadillo-protected app once with elevated permissions, it sets up the areas of the registry it needs and sets permissions there so that non-elevated accounts can henceforth work without seeing the "run as administrator" screen again. Yes, even completely different Armadillo-protected apps will thereafter be fine.

But… this screen is certain to cause some support headaches and phone calls. Especially if a user thinks he's already running as an administrator.

Workarounds

The easiest workaround is just to update to Armadillo 4.66 or higher. Armadillo's developers have found some "common" license storage that doesn't need to write to the common areas of the registry. I'm not certain whether this is as secure as when Armadillo also writes to the registry, though. So even though I am using 4.66, I'm still going to do the better workaround.

The better workaround involves an elevation trick. You'll recall that a process inherits the token of the process that launches it. And Vista realizes installation programs need to run elevated. So if there were a way to initialize Armadillo using the credentials of the elevated installer at the time it runs, the user wouldn't ever confront Armadillo's popup (Figure 22).

One of the Armadillo forum folks suggested this technique, and it's the approach I'm taking.

I've written a little do-nothing program to serve as the basis for elevating Armadillo's setup (actually, in my case it also does some other initialization functions, but it could be a simple Hello World without the hello world). I made a copy of the Armadillo project that protects my main program, and specified for it just to protect my do-nothing program In the copied project file, I turned down the protections (debugger-blocker, Soft-ICE detection, etc.) What's important if you're setting up any other Armadillo functions is that the Project ID be the same, as that's what determines which keys Armadillo creates and where it stores them.

My program is compiled with an asInvoker manifest, protected with the copied Armadillo project, then code-signed. I include it in my SetupBuilder project as a support file. At some point during the installation, I use the Run Program script item with the Wait for Program box ticked to run my dummy/setup program. Because of the manifest, Vista doesn't virtualize the registry calls. Because this small app is started from the elevated program, there's no user intervention required.

By the way...

In order to experiment with clean installations to get the Armadillo screen shot and information nailed down, I've installed Vista four times already this week on an ancient 550MHz Pentium3 with 512MB of memory – two times yesterday alone. (Talk about slaving over a hot stove!) The box doesn't have a bootable DVD drive, so I had to buy the installation CDs (there are *five*) from Microsoft. Welcome to the CD shuffle. But it's an easier installation than older versions of Windows. I've gotten it down to about an hour and a half from a bare drive (MBR re-initialized) to network drivers installed and box talking to the rest of my machines.

Conclusion

With the advent of Vista it seems to me imperative that Clarion developers link a manifest in all shipping applications, rather than risk any of the unpredictable side-effects of not having a manifest. And as I argued earlier in this series, it's especially important to sign any apps that may need to run elevated. UAC can be turned off altogether, or various parts such as Secure Desktop or virtualization of files and the registry can be disabled. But you can't assume your users will be running anything less than the default Vista security, and I feel it would be irresponsible to urge them to turn UAC down or off.

You'll need go through your apps and see where there's likely to be trouble. Don't plan on writing INI files or data files within C:\Program Files. Use CSIDLs to find where Microsoft wants you to write, or defy Microsoft and go your own way as I'm doing. Either way, try to be aware of the consequences for your users of earlier versions of Windows as well as for your Vista users. Separate anything that's going to require elevation into one or more utility programs distinct from your main app and link requireAdministrator manifests in those apps. Use the Vista elevation shield icon to warn users when they're going to need to give consent.

Play around with the virtualization stuff until you're pretty comfortable and confident that you'll have a fair chance of recognizing its meddling if some of your users experience strange bugs.

I used to tell my networking students that if computers were easy, a lot of us would be out of work. If nothing else, you may want to thank Vista and UAC for enhancing your job security.

Download the source

Additional Reading

- Administering Windows Vista Security: The Big Surprises: Mark Minasi's book is only 255 pages, and developers will learn a whole lot by reading a mere 150 of them or so. Do yourself a favor.
- Windows Vista Application Development Requirements for User Account Control Compatibility: An extremely valuable resource from Microsoft. If this link doesn't work, go to the main Microsoft web page and search for msdn windowsvistauacdevreqs.doc
- Microsoft web page explaining CSIDLs
- Another Microsoft CSIDL reference
- Friedrich Linder pointed out an MSDN forum thread on sharing data: (login required, but all you need to do is supply a user id and a non-verified email address)

Jane Fleming is a college dropout who subsequently lived four years in Europe, a year and a half in Mexico, and three years in India, and later taught yoga for a living in California (she’s been vegetarian

since 1970). She developed circuits and wrote assembly code for several embedded microcontroller projects during the 1980s. She began using Clarion Professional Developer for in-house projects back when Clarion was running display ads in InfoWorld and has used it very intermittently since. She is a former Microsoft Certified Trainer and taught Microsoft and Novell network administration at a business college for four years. Now widowed ten years, Jane plays classical piano and has found her métier as a semi-retired NRA-certified pistol instructor.

Reader Comments

Posted on Monday, May 14, 2007 by Peter Gysegem

This was a terrific series of articles Jane, thank you so much for adding light to this murky subject.

From within my SetupBuilder script, I am running the following command that removes virtualization from my program's registry key (a compiler variable, REGISTRYKEY, set to "SoftwareBeaver Creek SoftwareThe THERAPIST Basic". This avoids all the virtualization problems on that key.

```
FLAGS "HKLM[REGISTRYKEY]" SET DONT_VIRTUALIZE /s
```

I follow it up with a script function to grant access permissions on that same registry key. In testing, it appears to give me back the kinds of registry access my program had under XP. We haven't yet released the app so I don't know yet how well it will work in the wild.

Add a comment

Clarion Magazine

Synchronous Messages, SendMessage and SendMessageTimeout

by Larry Sand

Published 2007-05-11

In all the previous installments in this series (parts 1, 2, 3 and 4) I've sent interprocess messages using the asynchronous PostMessage function. These messages are sent without regard for what happens at the other end. But if you need a return value from the receiving window procedure you must use one of the two synchronous messaging functions: SendMessage or SendMessageTimeout. Both of these functions send the message to another window like PostMessage, however, when the receiving window is on another thread they wait for the window procedure to process the message and return a value. While they are waiting, they block the sending procedure until the window procedure returns.

SendMessage is prototyped like this:

```
CMAG_SendMessage(UNSIGNED hWnd, |
            UNSIGNED nMsg, |
            UNSIGNED wParam, |
            Long lParam),Long,Pascal,Proc,Name('SendMessageA') ,DLL(1)
```

And SendMessageTimeout is prototyped like this:

```
CMAG_SendMessageTimeout(UNSIGNED hWnd, |
                 UNSIGNED uMsg, |
                 UNSIGNED wParam, |
                 Long lParam,   |
                 UNSIGNED fuFlags, |
                 UNSIGNED uTimeout, |
                 *UNSIGNED lpdwResult),Long,Pascal,|
                                 Raw Name('SendMessageTimeoutA') ,DLL(1)
```

The prototype for SendMessage is nearly identical to that of PostMessage. The only difference is that PostMessage returns a BOOL and SendMessage returns a Long. BOOL is defined as a long integer for the Windows API; in Clarion the difference is in how you interpret the value. PostMessage returns zero for failure and non-zero for success. SendMessage returns the long integer that the called window procedure returns after processing the sent message.

SendMessageTimeout's first four parameters are identical to PostMessage and SendMessage. Those were described in the discussion of PostMessage. The fuFlags parameter controls the how the function times out and its blocking behavior see MSDN for complete documentation. The uTimeout is the timeout value in milliseconds, and lpdwResult is an UNSIGNED variable where the function places the return value from the window procedure. The return value of SendMessageTimeout is not the value returned by the window procedure it called (that's what the lpdwResult contains), it's just a result code, zero for failure and non-zero for success.

endMessageTimeout and WM_COPYDATA

You may remember that there was one message that the window procedure included in its Case uMsg statement but did not process. That was the WM_COPYDATA message. This message is designed to allow you to pass a pointer to some data; you can then copy that data into a variable or structure in the receiving process. If you were to simply pass the Address() of some variable to another process and try to read the data it pointed to, you would get a GPF or some other unpredictable result. The reason is that the two processes have different address spaces. You simply cannot pass a pointer to some data to another process without somehow marshalling the pointer. That's what WM_COPYDATA does for you. Behind the scenes it uses a memory mapped file. Memory mapped files allow your two processes to have a shared address space and this is what allows you to copy data between the processes.

The code for the final version of the InterprocessComs class is in the ipcCl.inc and ipcCl.clw files. First, look at the window procedure IpcScWndProc. The WM_COPYDATA message case originally did not have any code and now it calls the OnWmCopyData method, forwarding the wParam and lParam arguments.

```
Case uMsg
Of CMAG_WM_COPYDATA
  Return Me.OnWmCopyData(wParam, lParam)
...
```

OnWmCopyData is the new message handling method added to the class to process the WM_COPYDATA message. One important thing to note about the WM_COPYDATA message is that the pointers it passes are only valid while you're processing the message. As soon as your window procedure returns you cannot access the information anymore. Therefore, you must make a local copy while you're processing the message to have it available later on. The code for OnWmCopyData looks like this:

```
InterprocessComs.OnWmCopyData Procedure(UNSIGNED hWndFrom, |
                                Long pCopyDataStruct)!,UNSIGNED
cds     &CMAG_COPYDATASTRUCT
Result UNSIGNED,Auto
  Code
  If hWndFrom = Self.hWndPartner
    cds &= (pCopyDataStruct)
    Result = Self.ProcessCopyDataStruct(cds)
  Else
```

```
    Result = False
  End
  Return Result
```

OnWmCopyData introduces a new named group, the CMAG_COPYDATASTRUCT. This group is used by the WM_COPYDATA message to send your data to the other process. It's defined like this in Clarion:

```
CMAG_COPYDATASTRUCT Group,Type
dwData           UNSIGNED
cbData           UNSIGNED
lpData           UNSIGNED
                   End
```

The three elements, dwData, cbData, and lpData are used to describe your data to the receiving process. The first, dwData, is for your use and may contain any 32 bit value that you wish to send as part of the packet or as the only value. cbData is the size in bytes of the data pointed to by lpData. And lpData is the Address(), or, in the case of a threaded global in Clarion 6, the Instance() of the data you wish to pass to the other process.

The method declares the variable as a reference to the named group CMAG_COPYDATASTRUCT. You'll see how this is reference assigned to the value passed in pCopyDataStruct (the lParam) in a moment.

The handle to the source window is passed in the hWndFrom (the wParam) and it's compared to the hWndPartner property. If the sending window handle is not the same as the partner window handle you do not want to process the message. It's possible that some other application sent the message, and without prior knowledge of the structure of the data you cannot do anything. If the message did originate from your partner application then you need to reference-assign the pointer to the copy data structure passed in pCopyDataStruct to the cds reference variable. The parenthesis around the pCopyDataStruct causes the reference assignment to treat the data as a return value (you'll also see code in the Clarion documentation that adds zero to do the same thing. For example, the code shown in the documentation for NOTIFICATION has this assignment: "DM &= Q.ID + 0). This assignment statement only updates the address portion of the cds reference variable. You should treat the reference variable as read only; if you attempt to use a statement like Clear(cds) your program will GPF.

After the pCopyDataStruct is dereferenced, the ProcessCopyDataStruct method is called to do the actual copying. However, before I discuss the copy process I'm going to look at how the information got into the COPYDATASTRUCT and how it was sent to the window procedure.

The SendData method is responsible for filling in the elements of the COPYDATASTRUCT. Here's the code for the method:

```
InterprocessComs.SendData  Procedure(*String sData, |
                           UNSIGNED cbData)!,UNSIGNED
cds Like(CMAG_COPYDATASTRUCT)
  Code
  If Self.hWndPartner = 0
    Return False
  End
  cds.dwData = WMU_IPC_STRINGDATA
  cds.cbData = cbData
  cds.lpData = Address(sData)
  Return Self.SendWmCopyData(cds)
```

SendData accepts a String, sData, by reference and a UNSIGNED integer, cbData, that contains the size of sData in bytes. Then the method declares cds, a group of CMAG_COPYDATASTRUCT type. If this object has established a link with another InterprocessComs object there's a hWnd stored in the hWndPartner property. When there's another object to talk to the method fills in the elements of the cds group. First the WMU_IPC_STRINGDATA message is stored in the dwData element. This will be used to identify the contents of the packet to the receiver. The number of bytes pointed to by lpData is stored in cbData and the address of the string is stored in lpData.

After the cds group is filled, it's passed to the SendWmCopyData method. This method is responsible for sending the message to the receiving object in your partner process. This is the code for the SendWmCopyData method:

```
InterprocessComs.SendWmCopyData   Procedure(CMAG_COPYDATASTRUCT cds)!,UNSIGNED, Proc
smResult UNSIGNED
  Code
  If CMAG_SendMessageTimeout(Self.hWndPartner, |
                  CMAG_WM_COPYDATA, |
                  Self.hWnd, |
                  Address(cds), |
                  2, |  2=SMTO_ABORTIFHUNG
                  Self.SmTimeOut, |
                  smResult) = 0
    Self.TakeError(CMAG_GetLastError(), 'Sendmessagetimeout')
  End
  Return smResult
```

SendWmCopyData accepts a COPYDATASTRUCT as a parameter and is a thin wrapper around the Windows SendMessageTimeout function. SendMessageTimeout accepts the destination hWnd for the message, the WM_COPYDATA message, the wParam contains the hWnd of the sending window, and lParam contains the address of the COPYDATASTRUCT (cds). Next the SendmessageTimeout specific parameters are set. fuFlags is set to the constant two (2) for SMTO_ABORTIFHUNG (see MSDN for a discussion of other constants). SMTO_ABORTIFHUNG instructs the operation to time out if it detects a hung application. The timeout parameter is passed the object's smTimeout property. This property defaults to 1000 ms (1 second) and may be changed by calling the set and get SendDataTimeout methods. The local smResult variable is passed by address for the function to fill in with the return value from the other application's window procedure. SendMessageTimeout returns zero if there was an error, and the error code is passed to the TakeError method. If the call did time out, SendMessageTimeout sets smResult to zero, and the SendWmCopyData method returns the result of message processing.

When SendWmCopyData successfully sends the message, the receiving application processes the message via the OnWmCopyData method and the COPYDATASTRUCT is dereferenced and passed to the ProcessCopyDataStruct method as previously described. The ProcessCopyDataStruct method is responsible for copying the data from the pointer cds.lpData before the message returns.
The ProcessCopyDataStruct method is defined like this:

```
InterprocessComs.ProcessCopyDataStruct Procedure(|
          CMAG_COPYDATASTRUCT cds)!,UNSIGNED,Virtual
Result  UNSIGNED,Auto
  Code
  Result = False
  Case cds.dwData
  Of WMU_IPC_STRINGDATA
    If Not Self.sData &= Null
      If Size(Self.sData) >= cds.cbData
        CMAG_MemCpyRD(Self.sData, cds.lpData, cds.cbData)
        Result = True
        POST(WMU_IPC_STRINGRECEIVED)
      End
    End
  End
  Return Result
```

The sending application's SendData method set cds.dwData to WMU_IPC_STRINGDATA and ProcessCopyDataStruct checks to see if that's the value of cds.dwData. If it is then you've received your WM_COPYDATA message for the string. The class uses a String reference variable to point to the string buffer in the receiving program. This reference variable is set by calling the SetReceiveBuffer method. I'll discuss that in a moment; for now assume that Self.sData points to a string outside of the object.

If the string reference variable is not null and it's at least as large as the string being copied then the method makes a copy using the C library function MemCpy. MemCpy copies n bytes from a source memory location to a destination memory location. I've prototyped it like this:

```
CMAG_MemCpyRD (*? Dest, UNSIGNED Src, Long BytesToCpy)|
        ,Long,Proc,Raw,Name('_MEMCPY'),DLL(1)
```

This prototype allows you to pass the address from cds.lpData in the Src and Self.sData as the destination. The Dest parameter's *? (untyped by reference) data type with the Raw attribute instructs the compiler to only pass the address of Dest to MemCpy.

After the data are copied into the buffer pointed to by Self.sData, the WMU_IPC_STRINGRECEIVED message is POSTed to the target window's ACCEPT loop to notify it that data were received. The ACCEPT loop will listen for this event (message) and act on the data in the string buffer. You must keep the processing time to a minimum while processing the WM_COPYDATA message. Just copy the data, notify your ACCEPT loop something was received and return as quickly as possible.

You need a way to let the ipc object know where to assign the data, and that's the job of the SetReceiveBuffer method:

```
InterprocessComs.SetReceiveBuffer      Procedure(*String sData)
  Code
  Self.sData &= sData
  Return
```

As you can see, SetReceiveBuffer makes a simple reference assignment. One thing to note is that both the SetReceiveBuffer and SendData methods are prototyped to accept a String by reference. This isn't as restrictive as it may seem at first glance. Clarion groups are treated like a string and may be passed to these methods. You'll see how to put that to use in the upcoming WM_COPYDATA example.

That wraps up the implementation of the InterprocessComs class with its new ability to send string data between processes. Next time I'll show you how to use it in your programs, starting with sending a string between processes.

Download the source

Larry Sand is an independent software developer who began programming with Clarion in 1987. In addition to normal database development, he specializes in connecting Clarion to external devices like SCUBA diving computers, kilns, and satellite transceivers used in medical helicopters. In other lives, he sailed Lake Superior as the owner/operator of shipwreck SCUBA diving tours and later as a Master for the Vista Fleet. When Larry is not programming you'll find him messing about in boats, or with boats.

Reader Comments

Add a comment

Clarion Magazine

Coping With Vista's "Virtual" Reality, Part 1

by Jane Fleming

Published 2007-05-08

A major difference between how our society treats adults and how it treats children involves responsibility. We expect adults to be responsible for their actions. To varying degrees, depending on age, we cut kids a break.

Vista seems to have a similar mindset. However, while an attentive parent will try to correct a child's missteps, Vista may, like a parent in denial, pretend the errors never happened… with consequences that can be surprising.

I'm using this analogy to illustrate the difference between applications that Vista thinks "should know better" (i.e., they have a Vista-specific manifest, as I discussed in Coping With Vista - There's A Manifest In Your Destiny, Part 1) and apps that lack such a manifest. Vista has made substantial changes to who can access or write certain locations on the hard drive or in the registry. A Vista-aware app that violates those new rules will get slapped down. A non-aware app may continue to function, with Vista not returning errors but at the same time not behaving as expected.

For example, most applications install into the C:\Program Files folder by default. While many applications will use the logged-on user's My Documents folder to save data, many other apps save data to their own folders inside of Program Files. Oops. Vista has removed write-access to the Program Files folder (and by default, all of its child folders) unless the user is using an elevated token. (I discussed the split token and elevation in Coping With Vista - There's A Manifest In Your Destiny, Part 1). So, what happens when an application that doesn't know better tries to write on a Vista machine to some area Vista considers off limits? Smoke and mirrors. Stay tuned.

In the past, many applications used the Windows folder to store INI files. Some still write to WIN.INI. Clarion's default location for INI files is the Windows folder; there's a check box available in global properties to force the location to be the application's own folder instead. Guess what? Both of those are now off limits. Randy Rogers' recent ClarionMag article supplies a new INI class to get around this specific issue. (Personally, I've just been creating a Vista-aware INI filename in the Fuzzy Manager Init global embed. A kludge, but it works.)

Oh… and not that it's very common, but ordinary users can no longer write to the root of the C:\ folder, although they can create folders there.

Then there's the Windows registry. If you've ever run Regedit or RegEdt32, you'll have noticed that the registry is divided into sections (Microsoft calls them "hives"). Most of the registry information is stored in files that are located in the Windows\System32\config folder. There, you'll see files like SAM (Security Accounts Manager, responsible for local user accounts on the computer), SYSTEM, SOFTWARE, etc.

But each of us has an individual registry component as well, stored in our profile. On an XP machine, that component would be \Documents and Settings\UserName\ntuser.dat. When I log on to a computer, the instance of ntuser.dat in my profile is merged into the registry. This appears in Regedit as HKEY_CURRENT_USER. Each user has write permissions to this hive (unless that's been restricted by an administrator - all you have to do to restrict it for a given user is to rename his instance ntuser.man for "mandatory".)

Some applications store certain information in HKEY_LOCAL_MACHINE, which is the same data regardless of who's logged on. One example where there'd be a need to do so might be license information, or the path to which an update installer should install its files. XP with updated service packs requires an administrator account to write to HKEY_LOCAL_MACHINE. If you try to write to that hive as an ordinary user, you'll get a return code indicating that your write has failed. Vista also requires an administrator account - but an administrator account that's running elevated. If you are logged on as an administrator and *running an application that does not have a Vista-aware manifest* (and have not specified Run as administrator), you'll think you're writing to HKEY_LOCAL_MACHINE based on the return code. But you're not.

Smoke and mirrors

Microsoft has supplied a schematic drawing showing the architecture of User Account Control (UAC) (Figure 1). I discussed some items (Secure Desktop, Elevation Prompt, Installer Detection) in Part 2 of this series. In the bottom left Figure 1, there's an orange box labeled Virtualization.

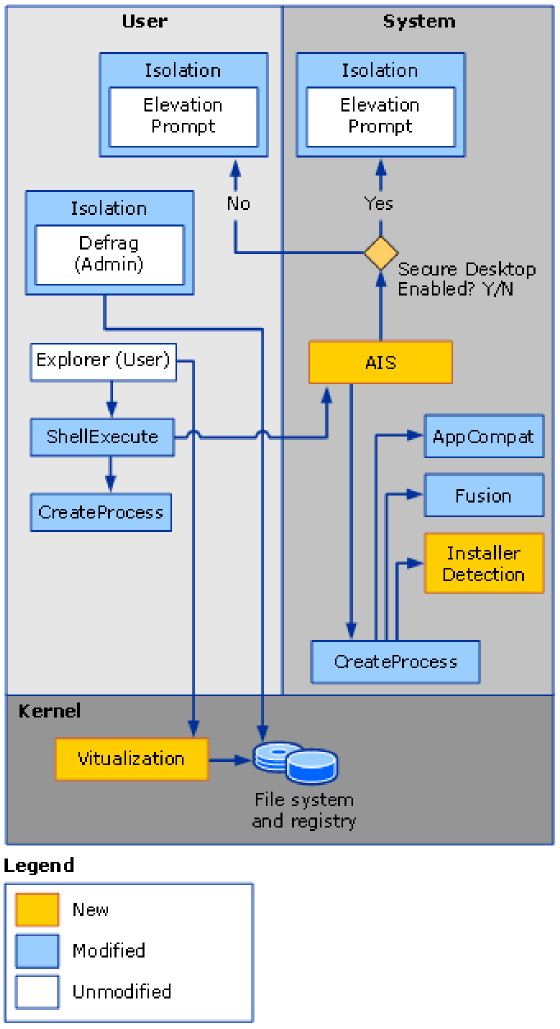


Figure 1. Microsoft drawing of the UAC architecture

Virtualization

Basically, "virtualization" as used by Vista means an application that doesn't have a Vista-aware manifest can continue writing to Program Files or HKEY_LOCAL_MACHINE or other protected areas and operate as usual. Actually, though, Vista is intercepting those write calls and placing the data elsewhere. You don't get an error, and you can read back the data you wrote. The problem arises if more than one user account is used on a particular computer.

I can think of three basic very generic computer-data-storage scenarios, one of which may easily trip over this new virtualization.

1. An individual using a computer, working with his own data. In this case, saving data to the personal My Documents folder (just called Documents in Vista) works fine. Each user has his own folder with adequate permissions.
2. An individual on a network, working with data on a server. Multiple users interact with the shared data on the server. This hasn't changed since earlier versions of Windows. No problem, assuming the network administrator has configured permissions correctly.
3. Various individuals working on a computer at different times using different user accounts, but who all modify the same data. This presents some challenges.

Prior to Vista, when various users on a computer required access to common data, having that data in a folder within Program Files was convenient. In that many older applications have gone that route, let's see what happens with Vista's file system virtualization.

Compile virtual.app from the download at the end of this article. It's a simple TPS browse/form. (**NOTE:** to run this app on a Windows 2000 machine, you'll need to comment out code in ThisWindow.Init in the Main procedure as indicated in the source. Be sure to uncomment the code again and recompile before running it on a Vista box. MSDN indicates that IsUserAnAdmin is available in SHELL32.DLL for Windows 2000, but Libmaker disagrees.)

Create a folder within C:\Program Files on an XP machine. Add some data. Log off and log on as a different user. You'll see the data you added. Add some more. Log off and back on as your first account. It's all there.

Now do the same on a Vista machine. Note than when you right-click Program Files and go to create a new folder, you'll get an elevation prompt. Do it and create the folder. Then copy virtual.exe and c60runx.dll, c60ascx.dll, and c60tpsx.dll into the folder. (I called my folder C:\program files\Virtual App, which may make figures 6 through 8 slightly confusing. Vista will virtualize the folder under these circumstances regardless of its name. You might want to name your folder C:\Program Files\MyApp)

Run the app and see what happens to data entered by different users. The following screen shots tell the story.

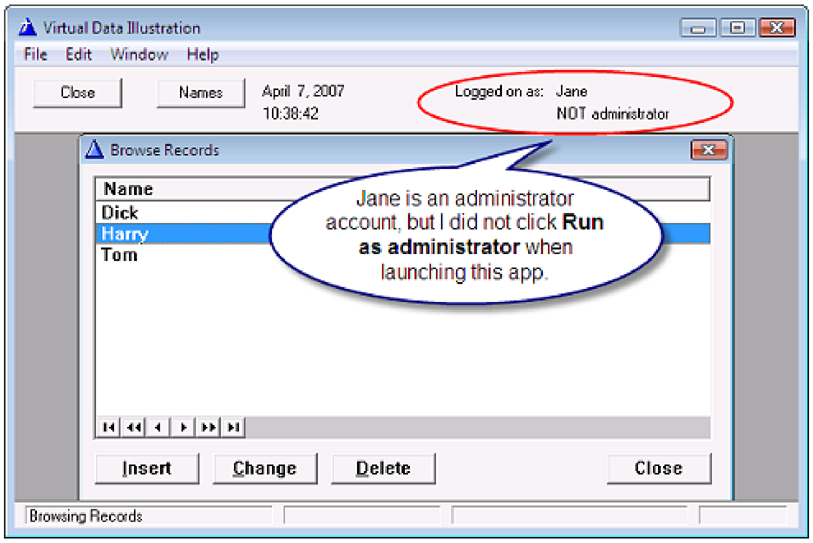


Figure 2. Logged on as user Jane, I create three records (view full size image)

In Figure 2, a check for whether a user is an administrator (using the API as in my app, or with VuFileTools) shows whether the elevated token is in use – not whether the user is a member of the Administrators group

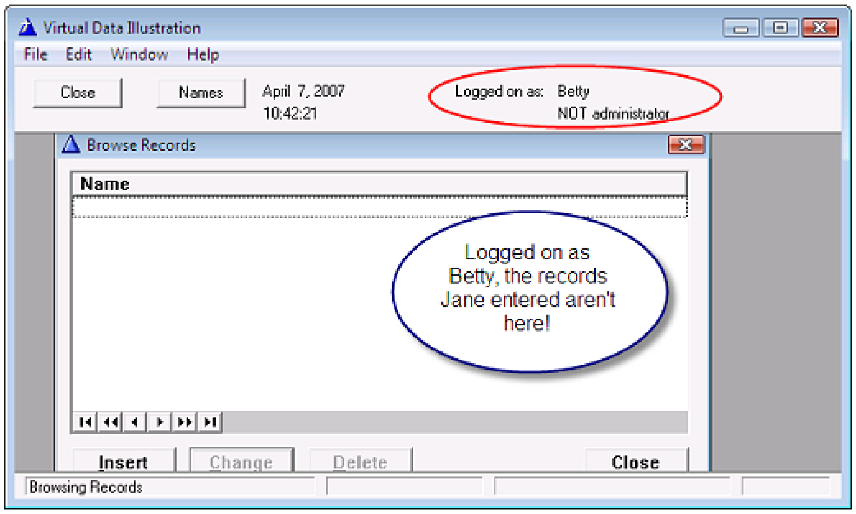


Figure 3. Betty's browse is bare! (view full size image)

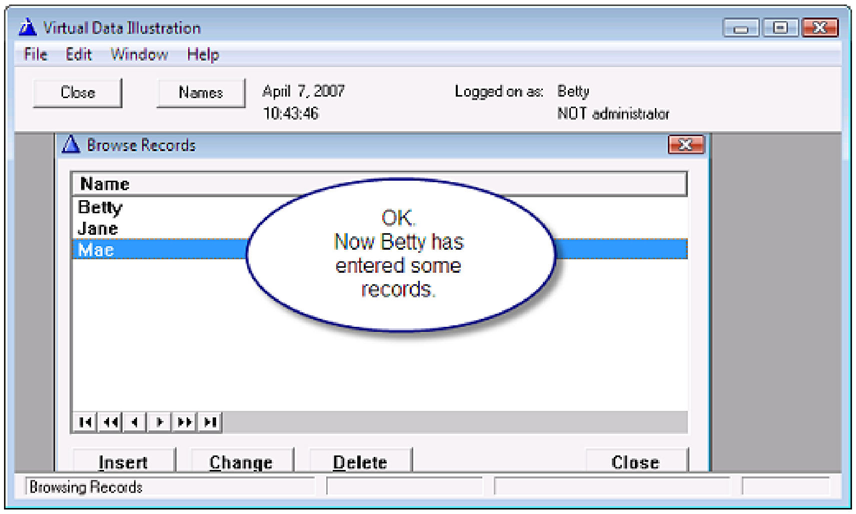


Figure 4. Betty enters some records. Will they show up when Jane is logged on? (view full size image)

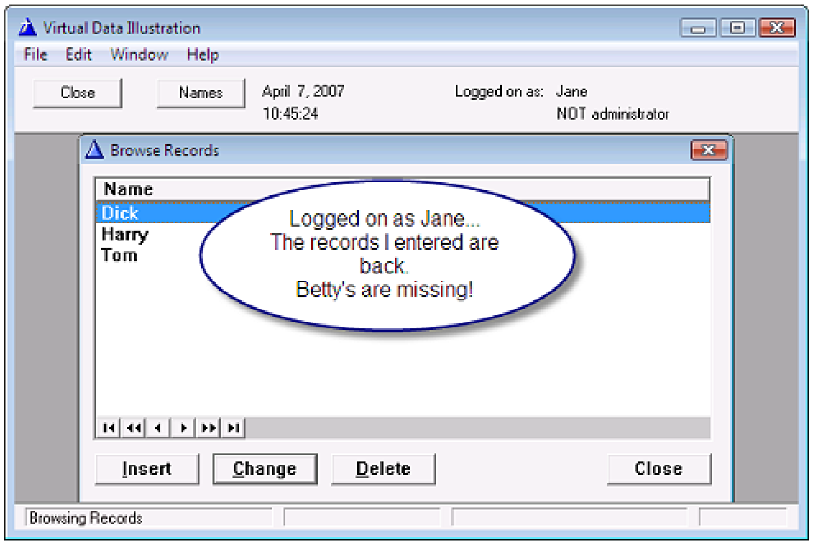
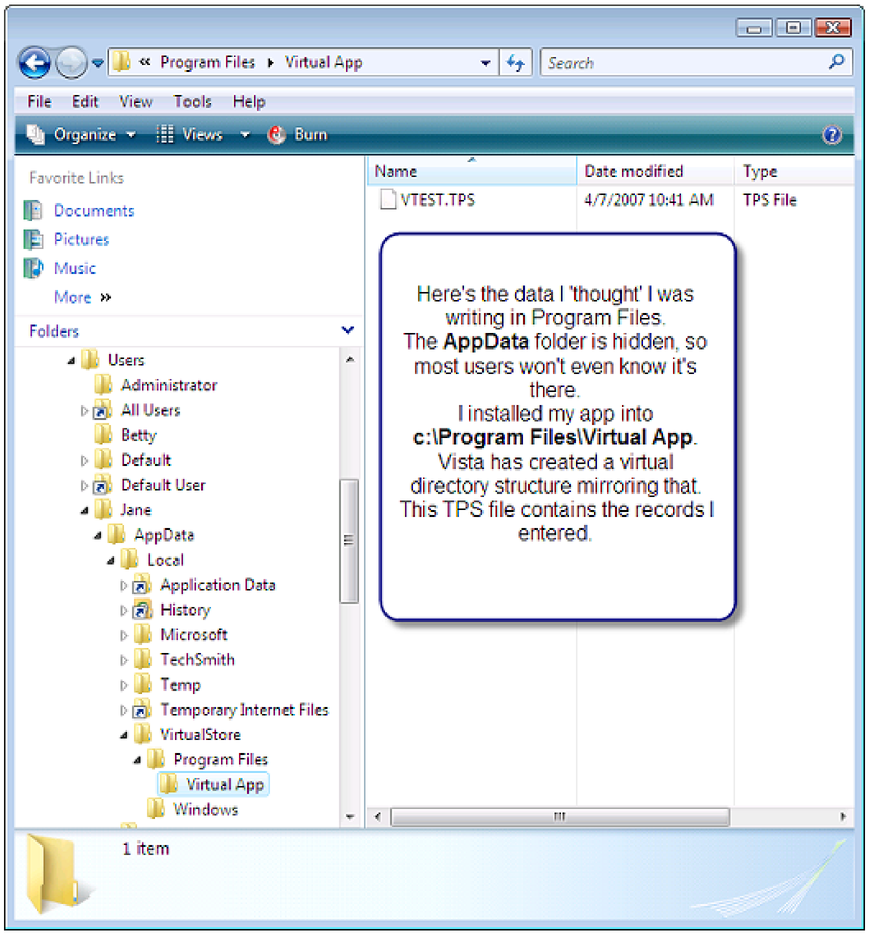


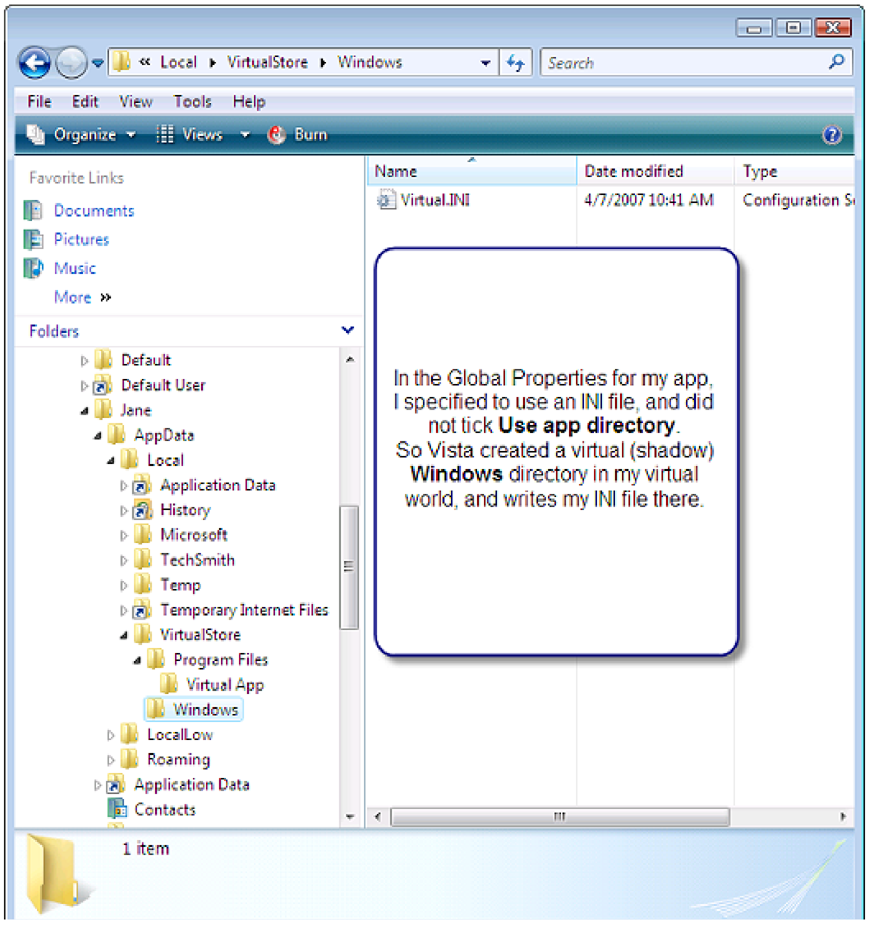
Figure 5. Nope! (view full size image)



Name	Date modified	Type
VTEST.TPS	4/7/2007 10:41 AM	TPS File

Here's the data I 'thought' I was writing in Program Files.
The **AppData** folder is hidden, so most users won't even know it's there.
I installed my app into **c:\Program Files\Virtual App**.
Vista has created a virtual directory structure mirroring that.
This TPS file contains the records I entered.

Figure 6. So *that's* where they went! Into a hidden folder within my user profile (view full size image)



Name	Date modified	Type
Virtual.INI	4/7/2007 10:41 AM	Configuration S

In the Global Properties for my app, I specified to use an INI file, and did not tick **Use app directory**.
So Vista created a virtual (shadow) **Windows** directory in my virtual world, and writes my INI file there.

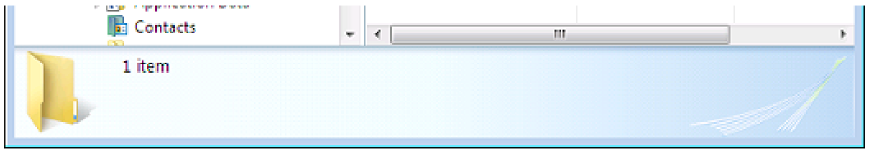
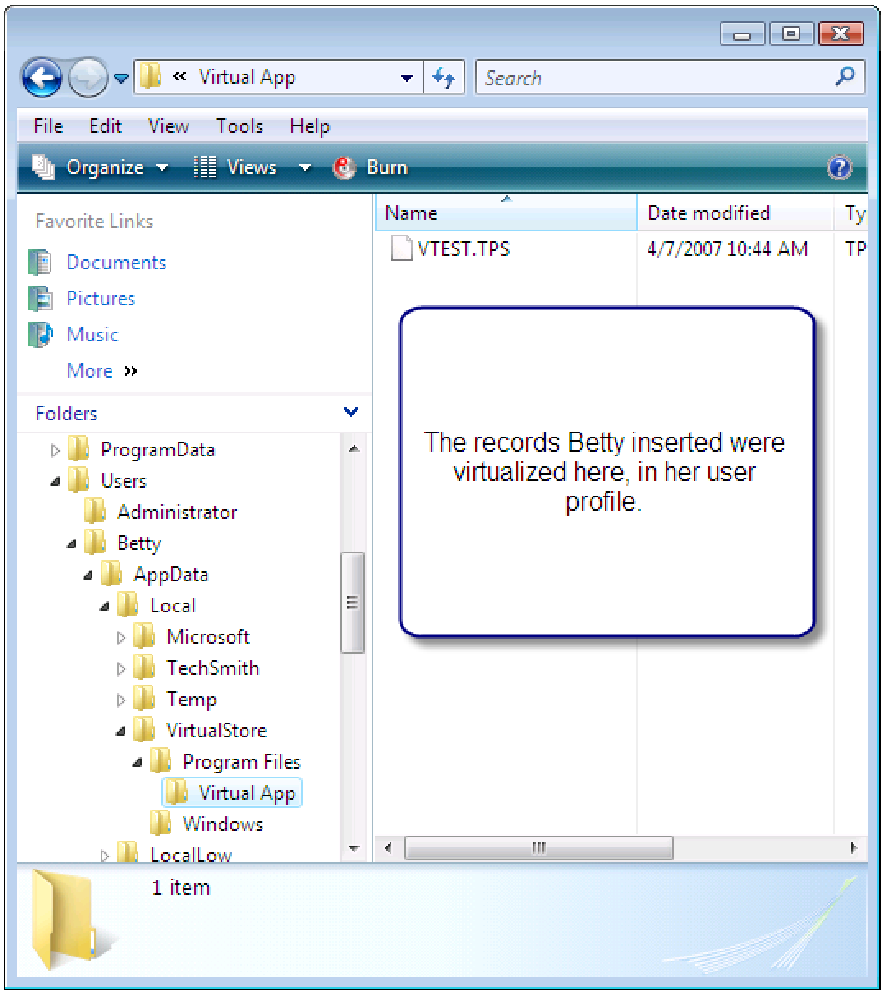


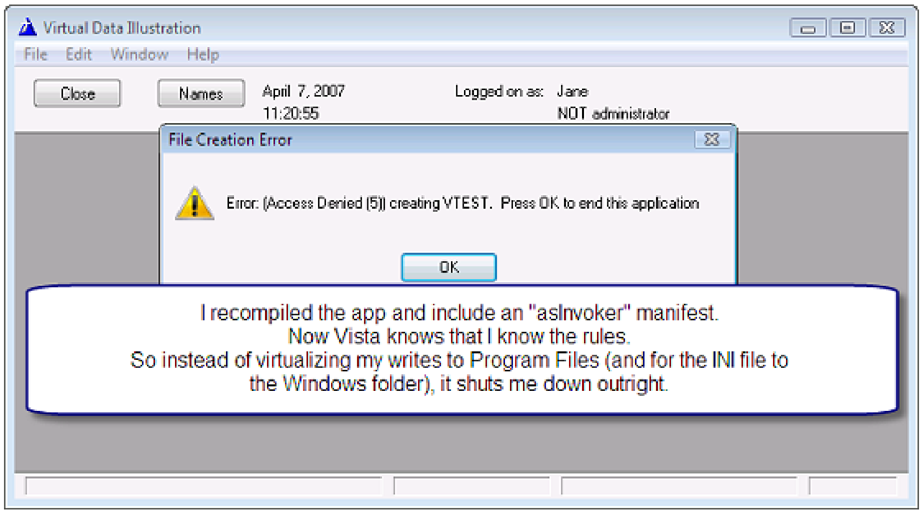
Figure 7. Each user has a hidden virtual Windows folder (view full size image)



Name	Date modified	Ty
VTEST.TPS	4/7/2007 10:44 AM	TP

The records Betty inserted were virtualized here, in her user profile.

Figure 8. Same thing for Betty in her profile



April 7, 2007
11:20:55

Logged on as: Jane
NOT administrator

File Creation Error

Error: (Access Denied (5)) creating VTEST. Press OK to end this application

I recompiled the app and include an "asInvoker" manifest.
Now Vista knows that I know the rules.
So instead of virtualizing my writes to Program Files (and for the INI file to the Windows folder), it shuts me down outright.

Figure 9. OK. Time to play with the big boys and include a manifest! (view full size image)

Strategy

It's probably best to do things Microsoft's way. You're less likely to run into trouble in the future, and it's a requirement if you want to get some kind of compatibility certification from Microsoft for your application.

That said, I'll explain in a minute why I'm *not* going to do things Microsoft's way. But first let's look at the "right" way.

For a number of years, Microsoft has had certain special environment variables.

For example, %systemroot% is a variable you can use in a batch file or with certain registry entries to find the folder in which Windows itself is installed. Try opening a command prompt and typing

```
dir %systemroot% [Enter]
```

Regardless of where your prompt's focus is, you'll get a listing of the Windows folder.

Microsoft now has a new system that they want us to use rather than those environment variables: Constant Special item ID List equates (CSIDLs). Randy Rogers' article includes a nifty little app that shows what those resolve to, and it's instructive to run his app on various operating systems.

Earlier inf this series I included a link to Microsoft's excellent article explaining UAC to developers. This article says that the kind of shared data I'm illustrating should be stored in a folder your application creates within the folder pointed to by CSIDL_COMMON_APPDATA

On my Windows 98SE machine, that resolves to

```
C:\WINDOWS\All Users\Application Data
```

On my Windows 2000 Pro machine, it resolves to:

```
C:\Documents and Settings\All Users\Application Data
```

On my XP Pro machine, it resolves to:

```
C:\Documents and Settings\All Users.WINDOWS\Application Data
```

On the two machines on which I'm running Vista, it resolves to

```
C:\ProgramData
```

(which is a hidden folder. Argh!)

My XP machine's path is not typical. I'm not sure after which repair or reinstallation or joining-and-leaving-a-domain it started adding that .WINDOWS stuff.

But this diversity of actual folder locations illustrates why Microsoft recommends using the CSIDL and setting any paths dynamically. This will mean being sure to use variables for the file names in your dictionary. And getting the CSIDL information to prime those file names.

Randy's class will take care of your INI path. Or you can kludge as I've been doing and put the code into the global FuzzyManager.Init embed after the parent call to initialize the INI file name.

If you don't need to deal with the INI file name, you can do your data file name initialization in the global Program Setup embed. The easiest way to see which global embeds get called when is to embed a few remarks in the ones you're interested in, click Project | Generate All, then switch to the Module tab and right-click the top app (which should say Default Program). Right-click on Default Program, select Module, then trace through the code and see where the embeds of interest fall. My choice of the FuzzyManager.Init embed has nothing to do with elegant OOPing; rather, it's just a convenient embed prior to where ABC initializes the INI file.

A Possibly Good Idea

If you're going to have program data paths that may vary according to the operating system and you expect users to back up their data (well, you back up *your* stuff, right?) it might be worthwhile to let the users know the data location. Consider adding an information window on your Help menu, or display the data location on your Help | About splash window. Of course, in that C:\ProgramData is hidden on Vista, that may not help your users much.

Why I'm Not Doing It Microsoft's Way

To anybody who knows me, it would no doubt come as a shock to learn that in my teens a shrink (their idea, not mine) told my parents that I have "insufficient respect for authority." That can't possibly be right….

There are two reasons why I've decided not to store my program data where Vista wants me to store program data: user-specific permissions and unnecessary complexity.

Setting permissions

First, there's the issue of permissions.

Create a new folder within C:\ProgramData on a Vista machine. Go into that folder and right-click, create a new text file, type a sentence or two and then save the file. Log off and log on as a different account (or just select Switch user). Now navigate to that file. Well, at least it's *there* this time instead of being virtualized. Try to edit the file and save it. Oops. You don't have adequate permissions.

Randy's article links to a good description of using the cacls.exe program to change permissions on folders you create. (Although cacls will run on a Vista machine, Vista tells you that it's now a deprecated command and asks you to invoke Icacls instead. Of course, Icacls won't work on previous versions of Windows). Friedrich Linder has announced that SetupBuilder 6.5 will have the ability to set permissions on folders, but my users sometimes (legitimately) copy the software folder to or from a USB flash drive rather than running the installation program.

Keeping it simple

Second, there's the complexity I'm not willing to foist on my users

I sell a scoring software package for pistol matches. It generally runs on a single computer. It may be installed "properly" using my installation package, or a user might copy it onto a USB drive and run it from there. I also tell users that they can just copy the contents of my data folder and e-mail it to another user, who can copy it into his data folder.

I don't want this to sound condescending, but the target audience I keep in mind is one particular police officer who drives a squad car in Indiana. He's not a computer guy. He's happy my software lets him do something he couldn't otherwise. To try to start training people like this gentleman to show hidden folders seems counterproductive and unnecessary.

There are also some occasions where my software needs to be networked to a shared data folder. In that case, having users figure out how to share hidden folders that have restricted permissions is not a tech support issue I crave.

So I'm cheating!

I have my app check whether it's running under Vista. If so, it checks to see whether the installation program has already set up its data folders the way I want . If not, it uses ShellExecute (as explained previously) to launch an elevated setup program that creates the folders and writes their paths into HKEY_LOCAL_MACHINE in the registry, where my app can find them regardless of which user might be logged on.

If you create a regular folder at the root of the C:\ drive, your users will have read/write access to it. I create a C:\MyStuff folder with several folders inside it and have my program use those folders for its data and INI files. Nothing is hidden. There are no complicated permissions issues. I'm not installing anything into a sensitive area that's a threat to the computer's security. Perhaps there's a bit more clutter than Microsoft would approve of, but that's life! (And I'm not looking to get my app certified by Microsoft.)

So why not just put my app into that folder as well?

In that the C:\MyStuff folder will be writeable by my users, why not avoid the whole C:\Program Files issue altogether and put my program there and let it create INI files and data files in that folder, just as many of us have put INI and data files in our program's folder in the past?

If you've ever had a machine infected by a virus, you may have seen how many executable files can get mutilated in a very short period of time. Any files in C:\Program Files or its subordinate folders cannot be modified on a Vista machine except by an elevated process. My program files would lose that protection if I placed them into C:\MyStuff.

So far I've discussed directory virtualization, but there's another area in Vista that gets virtualized, and that's the registry. More on that next time.

Download the source

Additional Reading

- Administering Windows Vista Security: The Big Surprises: Mark Minasi's book is only 255 pages, and developers will learn a whole lot by reading a mere 150 of them or so. Do yourself a favor.
- Windows Vista Application Development Requirements for User Account Control Compatibility: An extremely valuable resource from Microsoft. If this link doesn't work, go to the main Microsoft web page and search for msdn windowsvistauacdevreqs.doc
- Microsoft web page explaining CSIDLs
- Another Microsoft CSIDL reference

- Friedrich Linder pointed out an MSDN forum thread on sharing data: (login required, but all you need to do is supply a user id and a non-verified email address)

Jane Fleming is a college dropout who subsequently lived four years in Europe, a year and a half in Mexico, and three years in India, and later taught yoga for a living in California (she’s been vegetarian since 1970). She developed circuits and wrote assembly code for several embedded microcontroller projects during the 1980s. She began using Clarion Professional Developer for in-house projects back when Clarion was running display ads in InfoWorld and has used it very intermittently since. She is a former Microsoft Certified Trainer and taught Microsoft and Novell network administration at a business college for four years. Now widowed ten years, Jane plays classical piano and has found her métier as a semi-retired NRA-certified pistol instructor.

Reader Comments

Add a comment

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

Further Aussie DevCon notes

Direct link

Posted Monday, May 28, 2007 by Dave Harms

The 2007 Aussie DevCon has wrapped up, and there have been a number of newsgroup posts and blog entries that shed further light on SoftVelocity's presentations and the reaction of attendees. Bob's presentations, which covered the new IDE, Clarion 7, and Clarion.NET (in its WinForms, WebForms/ ASP.NET and Compact Framework incarnations, by all accounts left his audience both impressed and optimistic. Folks got to see the new data dictionary and visual data designer, both of which are (apparently) essentially complete and expected in the next Alpha release. The AppGen is still undergoing significant work and is not quite ready for alpha testing yet. Word out of the conference (which aligns with what I've heard from other sources) is that the bottleneck is getting the AppGen to talk nice with the rest IDE, with the additional info via Geoff Bomford that the .NET threading model is problematic:

> ...changes to the threading model by MS are still causing problems integrating the dictionary and designer into the app gen. Clarion is utilising a multi-threading technology for the app gen which is basically not supported by the single threading limitations of dot net. This has been a frustrating sticking point but Bob expects an imminent resolution to the problems.

Richard Bryce has posted his views on the conference. The turnout was about 60, which is a good number for that part of the world. Some key points from Richard's summary:

- The AppGen/embeditor were not shown
- All the visual designers, including ASP.NET and compact framework, were demo'd
- Adding PHP support isn't a huge step
- There will be a C# template chain at some point
- The report engine (at least in Win32) is the same beast

Richard didn't get a chance to ask Bob Z about plans for JSP support. It's worth noting that some years ago SV had a JSP product under development, but abandoned it as it was deemed to difficult to use. Certainly the Clarion.NET/ASP.NET product would mean a much shorter leap to JSP than the first time around, but I imagine the focus will be on .NET for the near future.

A C# template chain is a great idea, with one caveat: it can't (IMO) require SV runtime libraries. In other words, there's no sense in generating C# apps that rely on, say, the SV driver library or the report engine (unless those are open sourced, which I find highly unlikely). Yes, you'd get some C# business anyway, but really the point of having a C# code gen is so you can sell the product as not tying you down to a proprietary system. I'd expect the C# template to use ADO.NET or LINQ for data access. Reporting seems like more of a question mark.

Stu Andrews promises more coverage in the coming week. For now he says "There are a lot of words I could use to describe the past 5 days .. and “Awesome” would be every one of them." Stu is, of course, including the Capesoft World Tour in his kudos.

Kudos to Tony York and the gang for putting on the conference and giving the Aussies and friends a look at SV's latest. I haven't heard a peep yet out of SV regarding a US DevCon, but my guess is it's too late already for a fall DevCon, and my prediction is SV won't announce a DevCon until after AppGen is solidly planted in the new IDE and a Clarion.NET alpha or beta release is in the bag.

Aussie DevCon Clarion.NET report on the way

Direct link

Posted Saturday, May 26, 2007 by Dave Harms

The Aussie DevCon has just wrapped. I've received a report on Bob Z's closing session, in which he covered Clarion.NET including WinForms, ASP.NET, and compact framework apps. I've edited and send the report back for review, and hope to have it up on the site sometime on Sunday (wait, it is Sunday - okay, later today). Lots of cool stuff to report, and Bob's presentation appears to have been very well received.

Do you know your backups are good?

Direct link

Posted Tuesday, May 08, 2007 by Dave Harms

Mark Riffey points out this CIO Insight article on backup systems:

> A survey of more than 500 senior IT professionals reveals that a whopping 89 percent test their disaster recovery/failover systems only once a year or not at all, leaving their enterprises vulnerable to massive technology and business failures in the event of a disaster.

Just because you aren't a massive company doesn't mean your data isn't important. So you back up your data on a regular basis, right? But are you sure your backups are good? It's easy to get caught out, as Business 2.0 magazine recently discovered. Usually when I hear about a backup failure it's a tape-based system, but other storage systems are vulnerable as well.

I've started a new series of backup-related survey questions. This week's survey asks how often you test your backups.