CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Clarion Magazine

This edition includes all articles, news items and blog posts from April 1 2010 to May 31 2010.

Clarion News

Read 35 Clarion news items.

The ClarionMag Blog

Read 12 blog entries.

Articles

Source Code Library 2010.03.31 Available

April 12 2010

The Clarion Magazine Source Code Library has been updated to include the latest source. Source code subscribers can download the March 2010 update from the My ClarionMag page. If you're on Vista or Windows 7 please run Lindersoft's Clarion detection patch first.

Editorial: The Unofficial C7.x Gold Release

April 13 2010

It's a year since Clarion 7 officially went gold and was pretty much panned by the community. One year, many bug fixes and some important features later, is Clarion 7.1 gold-worthy? Clarion Magazine's editor and publisher says yes, it is.

Deriving The FileManager In A Multi-DLL App

April 19 2010

In the previous article in this series Dr. Parker looked at how to derive the FileManager to take control of CRUD operations. But he did not consider multi-DLL applications. In this article he corrects his oversight.

Browse Box Esoterica

April 20 2010

Browse boxes are a classic Clarion construct, dating back to the early days. And one very efficient way to interact with browse boxes, writes Steve Parker, is with the property syntax.

New Feature: The Clarion Roadmap

April 21 2010

One of the major features implemented on our new ClarionMag site is the Clarion Roadmap. Here's a sneak preview.

Alert(MouseLeft)

April 21 2010

Steve Parker continues his exploration of browse box properties and the unique way Clarion handles the left mouse click.

Why Unicode Matters To You (And SoftVelocity)

April 30 2010

In Part 1 Phil Carroll explained what Unicode is. In Part 2 he discusses why Unicode matters, and how Clarion could support Unicode with minimal disruption to existing applications.

What Is Unicode?

April 30 2010

In this first of two articles, Phil Carroll explains the weird and wonderful world of Unicode.

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Clarion News

View Wizard Version 7.02

ClarionTools is has released View Wizard version 7.02. This release is free to all registered View Wizard 7 developers. Existing customers are encouraged to update to this latest version. View Wizard supports ABC and Legacy Template Chains, for Clarion C55, C61, C62, C63, C70, and the latest C71 release. Changes include: Alternating Row Color (Greenbar) support with simple global activation; Added vwSortFieldMatch (STRING,STRING) function that is useful to determine if a locate is occuring on a specific field to further format output for the resulting locate; Window Resize extension template; Can now select where the column headings are displayed from; Column heading can be applied at runtime from the column heading in the dictionary or field definition; Can selectively turn off the dictionary options if desired; Picture format of the browse field can be applied at runtime from an override, or variable; Column freezing support that works mid-column; Can modify the alternate column description with static or runtime variable.

Posted April 8 2010 (permanent link)

PropertyGrid Wrapper 1.16

Version 1.16 of the PropertyGrid Wrapper template is available. Modifications include: Multi Threading enhanced; Category / Item ID variable increased; 'InPlace Buttons' added to Item definition; New method added - AddItemInPlaceButton; New method added - GetInPlaceButtonProperty; New method added - SetInPlaceButtonProperty. The new version can be downloaded from the Members area using the original download and registration details contained in your sales email.

Posted April 8 2010 (permanent link)

Clarion2Java On SourceForge

Clarion2Java is now a SourceForge project. The compiler and runtime are being used for a mid size application (350+ forms/browses/reports/etc) ported from Clarion 5.5, but the project it is still very young. The current file driver is heavily tuned for PostgreSQL. The project code is licensed under the Lesser LGPL, which means that you can use it, ship it and distribute without constraint. But if you make modifications to clarion2java itself - i.e. bug fixes to the compiler, those modifications must also be under an LGPL licence. Other licencing can be arranged on request if necessary.

Posted April 8 2010 (permanent link)

CHT 50% Off Subscription Drive

CHT is offering a "Target 100" offer at a discount of 50%. The 2010 "Target 100" order page will stay up until there have been 100 new subscribers who purchase this offer. At the end of the day on which the "Target 100" objective is reached (or exceeded) the page will come down.

Posted April 8 2010 (permanent link)

CHT Free C7 Batch Compile Generator 1.07

CHT's Free C7 Batch Compile Generator version 1.07 now posted for download. In this release the expiration limitation has been removed.

Posted April 8 2010 (permanent link)

Clarion.NET Training

Gus Creces is offering an on-line Clarion.NET Training Seminar series, based on a free digital book written by Charles Petzold. Gus will use .NET Book Zero as a bridge from the C-Sharp (C#) language into Clarion-Sharp (Clarion#).

Posted April 8 2010 (permanent link)

SV BLog On .NET AppGen

A new blog entry on the SV site indicates a .NET beta AppGen is 5-6 weeks away.

Posted April 12 2010 (permanent link)

Clarion2Java On SourceForge

The current release includes an executable demo called cookbook. If you have a postgres server nearby and java already installed, simply click on jar\cookbook-1.0.jar to run. Other changes in this release: Improved documentation; Includes a number of bug fixes in both the compiler and runtime libraries driven by feedback from a few people who downloaded and tested it; includes support for THREAD attribute.

Posted April 21 2010 (permanent link)

RPM Payment Options Include SWReg

There have been several users waiting for the ability to renew their subscriptions using a service other than PayPal. SWReg support is currently available only for renewals, but will eventually be available for new subscriptions. RPM features currently in beta: Extended folder & file variable lengths within RPM libraries; Centralized INI/Registry support within RPM "B" source module. Next, the back port to legacy, Clarion, template chain. Temporarily on hold: Support for Fomin Reports; Support for ReportDAT!; Support for ReportWriter 7. A new AFE installer will be available by the end of the month with an updated release of the FaxMan libraries. Work is continuing on the service version.

Posted April 21 2010 (permanent link)

DMC Gold 2.0.0.2

New in DMC Gold version 2.0.0.2 released April 15, 2010 for data transfers: Support for multiple linked conditions; Support for Manual Condition Builder (allows you to build the most complex conditions); Support for multiple linked functions; Support for linking a condition (or a "set" of conditions) to a function (or a "set" of functions); support for a function of a default value; Support for deformating a date or a time (enter values in readable format and let DMC work); Support for dynamically changing the value of a condition (create a condition with dates as "limits" & change those dynamically from an INI file). Two new videos are available.

Posted April 21 2010 (permanent link)

Clarion Third Party Profile Exchange - April 17 2010 Release

The Clarion Third Party Profile Exchange has been updated.

Posted April 21 2010 (permanent link)

Super Browse 7.01

Super Browse 7.01 is available for download. It's primarily a compatibility upgrade, making it work with Clarion versions 6.x through 7.1.

Posted April 21 2010 (permanent link)

Super Invoice 7.01

Super Invoice 7.01 is available for download. It's primarily a compatibility upgrade, making it work with Clarion versions 6.x through 7.1.

Posted April 21 2010 (permanent link)

Codejock 30% Discount

Motleysoft is offering a 30% discount on all Codejock ActiveX components and renewal subscriptions. To use it select a product, add to cart. After that use Coupon Code (enter it into field in Cart form) CLARION and push Update/Recalculate button. You will see a new discounted price (and small (E) icon).

Posted April 21 2010 (permanent link)

Clarion 7.1.7118 Released

Clarion 7.1 build 7118 has been released to CSP participants.

Posted April 30 2010 (permanent link)

PNG Resizing

Using Clarion 7, Capesoft Draw and Clarion FreeImage, you can create a very simple procedure to resize (and spit out PNG) your images.

Posted May 21 2010 (permanent link)

ClarionFolk Finished

Stu Andrews has posted his final entry on the Clarion Folk blog, but will be continuing to post Clarion-related items on his work blog.

Posted May 21 2010 (permanent link)

SetupBuilder 7.2 Release Announcement

Lindersoft has announced today the immediate availability of SetupBuilder Version 7.2. Version 7.2 includes several new and enhanced features to simplify and accelerate the development of reliable software deployments. Any organization planning to move its applications to the Windows 7 platform will be able to use SetupBuilder to comply with this new industry standard.

Posted May 21 2010 (permanent link)

Software Developers Profile Exchange - April 26 2010 Release

Included in this updated Software Developers Profile Exchange is extended and helpful information (data version) on Microsoft Partner Network Plans including - Microsoft Action Pack Subscription, Microsoft BizSpark, Microsoft DreamSpark, Microsoft Empower for ISVs, Microsoft Partner Programs, and Microsoft TechNet Plus.

Posted May 21 2010 (permanent link)

Version 2.05 of the TaskPanel Wrapper template is available.

Modifications include: New RemoveGroup method; New RemoveItem method; OCX Registration Enhanced; Template / Class Optimized; Multi Threading Enhanced; Fix for save/restore not working correctly when used on the AppFrame Extension Template; Fix for mimic properties and actions not working correctly when used on the AppFrame extension template. The new version can be downloaded from the Members area using the original download and registration details contained in your sales email.

Posted May 21 2010 (permanent link)

DevCon Gescla Brazil

DevCon Gescla Brazil will be held June 3-5 in Brazil in Belo Horizonte. Clarion Software Brazil, Urmet Daruma and FUMSOFT will be the sponsors of the event.

Posted May 21 2010 (permanent link)

TraceIt Updated

A new version of TraceIt is available. Just download the installer and use your serial number to update. Now Clarion 7.1 and Nettalk Web Server compatible.

Posted May 21 2010 (permanent link)

PDF-Viewer Update

Changes to the PDF-Viewer ActiveX class files include: OpenDocumentFromBLOB opens a PDF document stored in a BLOB field, which may be in a SQL DB or a TPS file; OpenDocumentFromHGLOBAL opens a PDF document stored in a Windows Global memory object (GMO) pointed to by the specified handle; OpenDocumentFromIStream opens a PDF document pointed to by an IStream pointer; The ExtractDocumentPages method extracts PDF pages pointed to by a Viewer Document ID and places them in a new Active Document, from which they may be saved or otherwise manipulated.

Posted May 21 2010 (permanent link)

Clarion2Java 1.2

Clarion2Java 1.2 supports Clarion 6, and includes an Clarion 6-generated demo app (musicdb).

Posted May 21 2010 (permanent link)

ClarionLive Lineup

ClarionLive has booked presenters into July. Click on the Coming Soon tab.

Posted May 21 2010 (permanent link)

SetupBuilder Competitive Upgrade For WISE Users

Symantec has announced the end-of-availability of WISE Installation Studio and WISE Installation Express. Symantec does no longer accept orders for new licenses of WISE Installation Studio or WISE Installation Express. In addition, Symantec does no longer accept renewals for maintenance agreements related to WISE Installation Studio. For users who want to move over from WISE Installation Studio to SetupBuilder, there is currently a special offer: switch to SetupBuilder 7 Developer Edition including a full 1-year maintenance and support subscription plan and pay only $245 instead of $399. This special offer expires June 30, 2010. Contact sales to save 40% off your move to SetupBuilder Developer Edition. Proof of ownership required.

Posted May 21 2010 (permanent link)

DMC 2.1.0.0

DMC Version 2.1.0.0 is now available. You can update to this new version from within DMC or you can use the full installer. Changes include: Menu with keyboard shortcuts for more user friendly IDE; Global setting to hide / unhide the left navigation bar - entirely resizable frame (smaller than 800x600) is supported. SQL views are available as source and in viewer with their definition as declared in the SQL server (new tab); Create an SQL VIEW from a manual query and Export the Structure to TXD / DCTX; In Projects, a button to repair all when master key no longer works; Full support for Ms SQL SCHEMAS (displayed as dbo.TableName etc ...); Full support for Excel 2007 /2010 files and illegal characters in column names (spaces and accentuated letters); Support for Oracle - Postgre SQL and Ms SQL schemas in cloning task (allows you to clone a table from schema a to schema b); mapping option, function lookup on external table to act as filter.

Posted May 21 2010 (permanent link)

Strategy Online Updates

Strategy Online has launched a new page with news pertaining to releases, product development, and general company info and updates.

Posted May 21 2010 (permanent link)

CJ Popup Control Demo App

The soon to be released updated Noyantis Suite Controls Template will include the CJ Popup control and this demo will give you a chance to learn how to manually control, build and play with popup windows. The Template will make the control much easier to implement and to track events. The demo app allows you to see how the various properties interact with the CJ Popup control. You can use it to see how to create Themed Popups, custom Popup Logos and how to use XAML to create custom Popup windows. It uses the CJ MarkupLabel & Popup controls from the Suite Controls OCX and the Skinframework control via the Noyantis Template and they are required to compile the demo app. It is a CW 6.3 app but will convert to CW 7+ without any changes.

Posted May 21 2010 (permanent link)

RPM build 6697 for 7.1.6695

RPM build 6697 for 7.1.6695+ is available. In addition to the RPM Dummy Report fix and improved Office Inside support added to build B, this build adds better user option and file support. All file folder and name variable lengths have been extended, in some cases significantly. INC files are now used for the option group TYPE definitions and centralize INI/Registry functions have been added to the "B" source module. A C6.3.9051+ build should be available shortly. In process is the back port to legacy and added support for Fomin, ReportDAT! and ReportWriter 7 (as soon as it arrives). The next major addition to RPM is SQL report archiving which is already in design. For AFE users a new server installer should be ready around the end of May or early June.

Posted May 21 2010 (permanent link)

EasyExcel 4.05

EasyExcel 4.05 is now available. New in this release: Libraries were recompiled for CW 7.1 (7075); Fixed an error message in the Init template in Clarion 7.x; Various changed and new methods; SetFormat code template added AutoFormat parameter; List2Excel code template added an ability to export all records of the browse list (not only visible).

Posted May 21 2010 (permanent link)

FinalStep 2.24

FinalStep 2.24 has been released. Changes include: A new installer (requires new serial), all revised for compatibility with Clarion 7.1; Pseudo maximize, now removed from the main template and separated on a separated template (for both legacy and ABC), and reprogrammed for better precision on windows resizing.

Posted May 21 2010 (permanent link)

PrintWindow 1.25

PrintWindow 1.25 has been released. Changes include: A new installer (new serial required) ready for Clarion 7.1; Report title now takes a variable.

Posted May 21 2010 (permanent link)

KSpng 1.3

Version 1.3 of the KSpng library is available. List of features in the latest version:; Display of the graphic image on Clarion window and report including .TIFF, .PNG and .EMF files in addition to standard Clarion picture formats; Alpha transparency channels support; URL support for file names; Conversion between various picture formats and grayscale support; Saving content of an image control to graphic file; Support for the linked resources (Graphic files can be linked to any exe or DLL); Clipboard support (Load and save graphic image file to and from clipboard); Screen capture to graphic format (Desktop, window or rectangle area). No other third party products are needed. Just one 25K DLL. Clarion 5.5, 6.3 and 7.1 both Legacy and ABC are supported. Demo version available.

Posted May 21 2010 (permanent link)

EasyNaviBar 1.01

EasyNaviBar 1.01 has been released. Changes include new GetVisible / SetVisible methods. This is a free upgrade for all customers who have a current (valid) subscription plan.

Posted May 21 2010 (permanent link)

CLARION MAGAZINE
READ. LEARN. SOLVE.

The ClarionMag Blog

April 30 Update

We're getting close to release! Some of the features in this latest update:

- UI changes, inculding favicons for news items
- You can now rate articles - please do so!
- The Clarion Roadmap is now online! Check it out, including the ability to create your own personalized view of ClarionMag's content.

Posted April 30 2010 (permanent link)

Launch scheduled for today!

The launch of the new ClarionMag site is scheduled for today, May 19 2010!

Posted May 19 2010 (permanent link)

We're live!

And we're live! Or we should be - the DNS changes me take a few minutes to make their way out into the ether.

Posted May 19 2010 (permanent link)

About the news page

Until the news page is done, please just click on the links in the news items list on the home page. Thanks.

Posted May 19 2010 (permanent link)

Customizing the text color

If you're not a fan of the dark gray we're using for the base text color, you can change it. Just log in to the update form and look for the "Default text color" field. Enter a valid HTML color (such as #000000, for black) and save.

Posted May 20 2010 (permanent link)

Login bug - checking it out now

It's always the little things. There's a problem right now with the login page when you supply the wrong password. Checking it out now.

Posted May 20 2010 (permanent link)

Logon bug fixed

The logon bug has been fixed, and you should now get an error message when you try to log on with invalid credentials.

Posted May 20 2010 (permanent link)

Another logon fix

The "remember me" functionality on the logon page had, as it turns out, a very limited memory. Like 30 minutes, on account of some custom code that ... oh, who cares. It was my fault.

The default auth cookie retention time for ASP.NET MVC, it appears, is 48 hours, which is a tad low as well. So I've set up a 60 day rolling expiry, which means the cookie expires 60 days after your last visit.

Posted May 21 2010 (permanent link)

RSS feeds returning soon

A few folks have noticed that RSS feeds are not working at present. Like the news page, they will be functional in the very near future.

Posted May 24 2010 (permanent link)

Printer Friendly is back

I've posted an update which adds "printer friendly" page support to articles.

Posted May 25 2010 (permanent link)

Email id / password corrections

I inadvertently wrecked a couple of pages yesterday when I added the printer-friendly support, including the link to retrieve your password and/or user id. Please note that this page still refers you back to the old site, so remember to come back here when you're done.

The new password / id retrieval page is currently #2 on my hit list, so it should be done today.

Posted May 26 2010 (permanent link)

RSS feeds are back!

The RSS feeds are back! For a list of available feeds, see the feeds page.

Posted May 27 2010 (permanent link)

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

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.

Subscribe to the source code library now

What isn't changing

Source code will still be available via individual articles. Nothing's going away. This is an added feature, not a replacement for how you get source code now.

Pricing

See the Clarion Magazine store for current pricing.

Subscribers get the source code library at a discount. If you wish to subscribe to Clarion Magazine and buy the source library subscription at the same time, the store will give you the magazine subscriber pricing on the source library.

What do I get?

When you complete your purchase you'll receive a confirmation email telling you to log in to My ClarionMag where you'll find your link to the source library download. The zip file contains a signed, SetupBuilder-powered installer which will set up the source code library on your computer.

Hundreds of source files and examples

At present there are well over 300 zipped source examples included in the library. The download is in excess of 30 megabytes.

Terms of Use

As a subscriber to the Clarion Source Code Library you are entitled to use the source code and examples without restriction in your own applications. You may not, however, distribute the source code or use it to produce third party products without the permission of CoveComm Inc, publisher of Clarion Magazine.

Upgrades

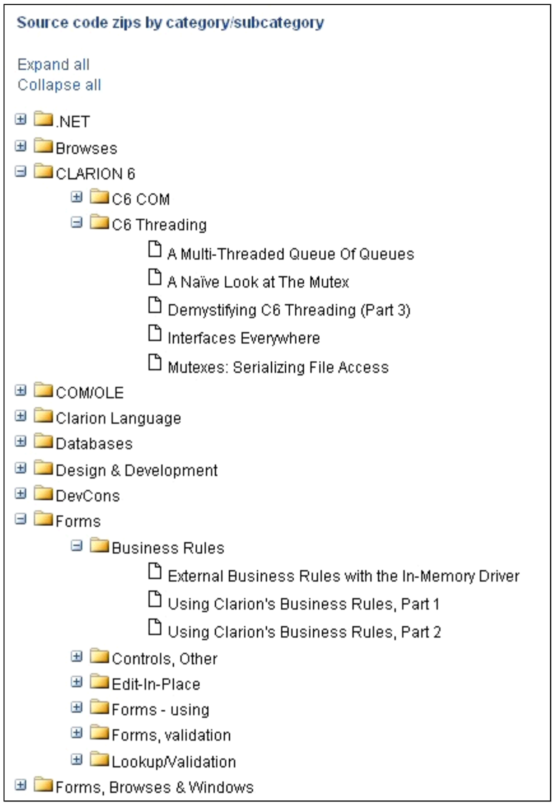
You are entitled to free upgrades as follows:

- for one year from the date of purchase, **or**
- as long as you have a current Clarion Magazine subscription

Log in to My ClarionMag to download the latest release.

Tree navigation

In addition to the full list of articles the source code library now comes with a categorized tree view of the articles (partial screen shot shown below):



Source code zips by category/subcategory

Expand all
Collapse all

- ⊞ .NET
- ⊞ Browses
- ⊟ CLARION 6
  - ⊞ C6 COM
  - ⊟ C6 Threading
    - A Multi-Threaded Queue Of Queues
    - A Naïve Look at The Mutex
    - Demystifying C6 Threading (Part 3)
    - Interfaces Everywhere
    - Mutexes: Serializing File Access
- ⊞ COM/OLE
- ⊞ Clarion Language
- ⊞ Databases
- ⊞ Design & Development
- ⊞ DevCons
- ⊟ Forms
  - ⊟ Business Rules
    - External Business Rules with the In-Memory Driver
    - Using Clarion's Business Rules, Part 1
    - Using Clarion's Business Rules, Part 2
  - ⊞ Controls, Other
  - ⊞ Edit-In-Place
  - ⊞ Forms - using
  - ⊞ Forms, validation
  - ⊞ Lookup/Validation
- ⊞ Forms, Browses & Windows

Topical tree view of articles with source code, several categories expanded

The articles listed in the tree view are links to the article summary (below, in the same document). The summary also has links to the source code (which is on your hard disk) and the original article on the ClarionMag web site.

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Editorial: The Unofficial C7.x Gold Release

By Dave Harms

Posted April 13 2010

One year ago, on April 13, 2009, SoftVelocity released the Gold version of Clarion 7.0.

Now, given that software is never done and complex software is never bug-free, Gold is a subjective label at best. But I don't know anyone outside of SoftVelocity who gave credence to that release as a Gold product. There were far too many bugs, the IDE was unstable, and the product was missing important functionality present in C6.3 (such as the window previewer).

Since "gold" release we've had twelve months of grumbling, bug fixing, testing, crashes, successes, general misery, and occasional elation. I'm sure it's not just Clarion developers wishing they never see another year like that one.

But that year has passed, and a great deal *has* changed. The IDE, while not perfect, is generally stable. Most of the really big, bad bugs have been squashed. Missing functionality has, for the most part, been restored.

More and more I hear about Clarion developers porting their apps from C6.3 to C7.1 and not looking back.

So are we there yet?

Yes, I think we are.

Of course, the usual disclaimers apply. You still have to do your own due diligence and decide for yourself if C7.1 meets your requirements. Creating new apps in C7 is undoubtedly the safest path. And I can't tell you in good conscience that you won't have any problems at all converting your C6.x apps to C7.1, because you almost certainly will. It's pretty much unavoidable, given that at least some of C7.x's troubles are due to long-standing bugs in C6.x.

But it *is* different now.

I've been racking my brain for a good metaphor for what I see happening. Has C7.1 reached critical mass? I trust not - that term has an unpleasant meaning in software development.

Okay, how about escape velocity? Is 7.1 breaking free of bad bugs and dashed hopes? Yeeessss. Ish. Or maybe it's a sea change, or the tide is turning. Or we're over the hump. Or it's not just the light at the end of the tunnel, it's the actual end of the tunnel.

Ah, the tunnel. What better way to describe the last year? (Or two. Or three.) Clarion 7.x, chugging along underground, uphill most of the time, through dank, seemingly endless, darkness. And then the light in the distance, appearing, disappearing, appearing again, a faint, hopeful glow.

I must have fallen asleep toward the end and I missed the tunnel exit, so I'm not entirely sure when we made that transition. Was it build 6955? 7104? Honestly I'm still a little groggy from the experience, so you'll forgive me if I just wait a while and let the sun soak into my bones. I'll be up for a little celebration in the observation car later.

Meanwhile, welcome to the surface. Welcome to the unofficial gold release(s).

And enjoy your time here. But please, please if you have a problem with C7.x, post a bug report in the PTSS, and make it public if at all possible. There's still work to be done. There's always work to be done.

David Harms is an independent software developer and the editor and publisher of Clarion Magazine. He is also co-author with Ross Santos of Developing Clarion for Windows Applications, published by SAMS (1995), and has written or co-written several Java books. David is a member of the American Society of Journalists and Authors (ASJA).

Article comments

---

by Mark Riffey on April 13 2010 (comment link)

I'll feel better about it when the ?Unknown thing is done. It doesnt happen to me, but it happens to enough folks that Im concerned about losing work. Any other concerns I have you can find in the PTSS:)

---

by Dave Harms on April 13 2010 (comment link)

Thanks Mark - definitely that's one that would be good to see fixed.

Dave

---

by Paul MacFarlane on April 14 2010 (comment link)

Actually I think it was a good year - and dedicated Clarionites wanted it. Making it Gold forced us to use it, test it, break it. We'd have to go through that sometime anyway. The sooner the better!

Now it looks like more resources are moving to Clarion# and the new template engine as 7.x gets polished. More fun ahead !

---

by Dave Harms on April 14 2010 (comment link)

Paul,

We have different definitions of "fun"<g>. But definitely more actual fun ahead is a good thing.

Dave

---

by Donald Ridley on April 21 2010 (comment link)

Good article Dave. I've kind of come into the game late. My first experience with C7 was 7.1 build 7014. So far I can honestly say I like it. It is unstable. It has some quirks. But I have been able work through them. In fact, someone mentioned a compatability problem with C7.1 and the ATI Catalyst Control Center. Once I got rid of ATI my C7 problems mostly went away.

Keep up the good work Dave!

by Dave Harms on April 22 2010 (comment link)

Thanks, Donald!

Dave

by Robert Hutchison on April 27 2010 (comment link)

Spent all day today with 7.1 build 7075 it is a lot better for sure, but I don't think it is 7.1 or Gold its still pretty rough and Beta as far as I am concerned. I wish I had the time to report all the bugs to Soft Velocity but I don't. I look forward to the day it is stable enough to use.

by Dave Harms on April 27 2010 (comment link)

As noted, not everyone will see the current releases as gold-quality. Thanks for the data point.

Dave

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

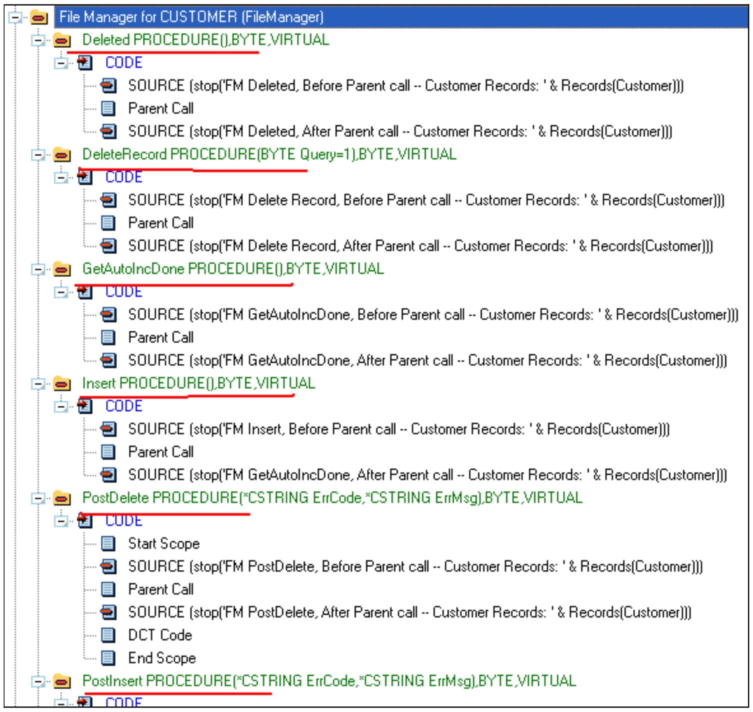
Deriving The FileManager In A Multi-DLL App

By Steven Parker

Posted April 19 2010

In the last action packed episode, "He who may not be named," editor of the world’s most widely read Clarion publication, thought "It might be worth investigating/explaining the difference between one-off embed code changes and deriving the `FileManager` or `RelationManager` (I've done neither recently)."

I discovered that there are a number of global embeds related to CRUDdy operations. Indeed, I discovered that there are all manner of global `FileManager` embeds and they surround virtually every step of every Insert, Update or Delete operation, plus many other things.



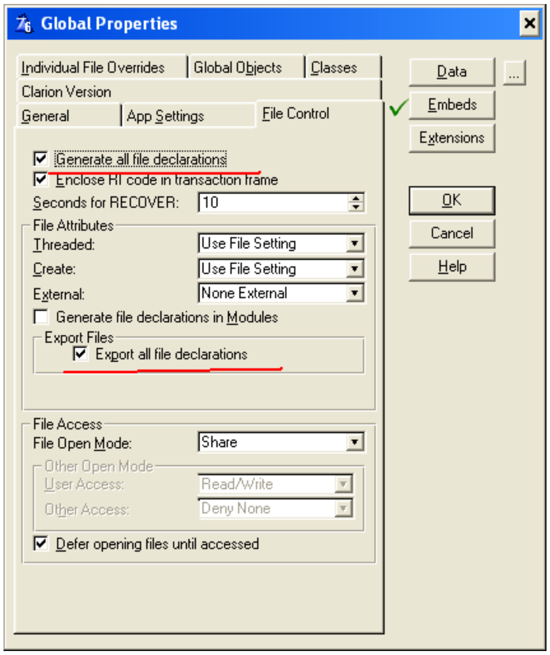
```
File Manager for CUSTOMER (FileManager)
  Deleted PROCEDURE(),BYTE,VIRTUAL
    CODE
      SOURCE (stop('FM Deleted, Before Parent call -- Customer Records: ' & Records(Customer)))
      Parent Call
      SOURCE (stop('FM Deleted, After Parent call -- Customer Records: ' & Records(Customer)))
  DeleteRecord PROCEDURE(BYTE Query=1),BYTE,VIRTUAL
    CODE
      SOURCE (stop('FM Delete Record, Before Parent call -- Customer Records: ' & Records(Customer)))
      Parent Call
      SOURCE (stop('FM Delete Record, After Parent call -- Customer Records: ' & Records(Customer)))
  GetAutoIncDone PROCEDURE(),BYTE,VIRTUAL
    CODE
      SOURCE (stop('FM GetAutoIncDone, Before Parent call -- Customer Records: ' & Records(Customer)))
      Parent Call
      SOURCE (stop('FM GetAutoIncDone, After Parent call -- Customer Records: ' & Records(Customer)))
  Insert PROCEDURE(),BYTE,VIRTUAL
    CODE
      SOURCE (stop('FM Insert, Before Parent call -- Customer Records: ' & Records(Customer)))
      Parent Call
      SOURCE (stop('FM GetAutoIncDone, After Parent call -- Customer Records: ' & Records(Customer)))
  PostDelete PROCEDURE(*CSTRING ErrCode,*CSTRING ErrMsg),BYTE,VIRTUAL
    CODE
      Start Scope
      SOURCE (stop('FM PostDelete, Before Parent call -- Customer Records: ' & Records(Customer)))
      Parent Call
      SOURCE (stop('FM PostDelete, After Parent call -- Customer Records: ' & Records(Customer)))
      DCT Code
      End Scope
  PostInsert PROCEDURE(*CSTRING ErrCode,*CSTRING ErrMsg),BYTE,VIRTUAL
    CODE
```

Figure 1. Global FileManager CRUD embeds

Starting with the Tutor.APP, from the 6.3.9056 Examples, I created an app in which I placed

STOPs in every embed that had anything to do with CRUD operations for the `Customer` file, both before and after the `Parent` call. That demo showed just which, where and when various `FileManager` methods were called.

In the last episode, I did not consider multi-DLL apps. I did speculate that overriding the `FileManager` in the data DLL would be all that was required. This was actually an assumption on my part. Because the data DLL generates and exports all file definitions



☑ Generate all file declarations
☑ Enclose RI code in transaction frame
Seconds for RECOVER: 10

File Attributes
Threaded: Use File Setting
Create: Use File Setting
External: None External
☐ Generate file declarations in Modules

Export Files
☑ Export all file declarations

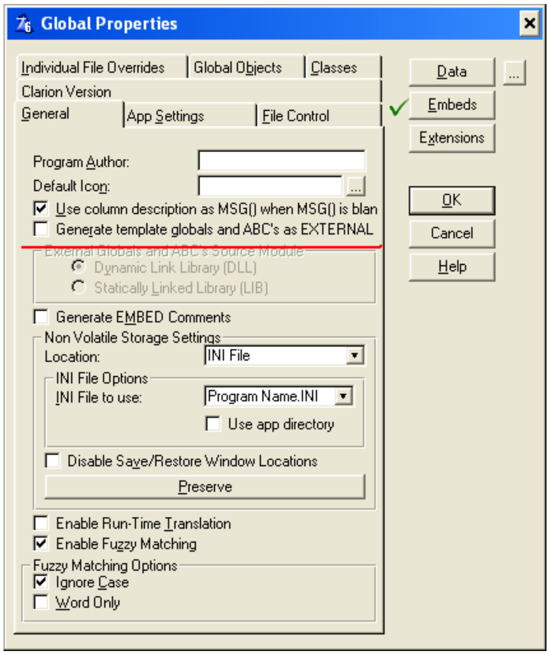
File Access
File Open Mode: Share

Other Open Mode
User Access: Read/Write
Other Access: Deny None

☑ Defer opening files until accessed

Figure 2. Data DLL exports all file definitions

and compiles in all ABC objects



Program Author:
Default Icon:
☑ Use column description as MSG() when MSG() is blan
☐ Generate template globals and ABC's as EXTERNAL

External Globals and ABC's Source Module
● Dynamic Link Library (DLL)
○ Statically Linked Library (LIB)

☐ Generate EMBED Comments

Non Volatile Storage Settings
Location: INI File

INI File Options
INI File to use: Program Name.INI
☐ Use app directory

☐ Disable Save/Restore Window Locations

☐ Enable Run-Time Translation
☑ Enable Fuzzy Matching

Fuzzy Matching Options
☑ Ignore Case
☐ Word Only

Figure 3. ABC's are not external to the Data DLL

and the `FileManager`s and `RelationManager`s are ABC objects and because those objects are created by the templates ... I just assumed that they would be exported along with the rest of the template globals and ABC classes and objects. Therefore, I just assumed that if I put code in the CRUD embeds in the data DLL’s `FileManager`s, that code would work in apps referencing this data DLL.

It ought to work. But does it? It's time to find out for certain.

The data app

The data app is simple. To create it, I just have to tick the "Generate all file declarations" (Figure 2), "Export all file declarations" (also Figure 2) and make sure that "Generate template globals and ABC's as EXTERNAL" is unchecked (Figure 3) – this ensures that they are compiled into the DLL.

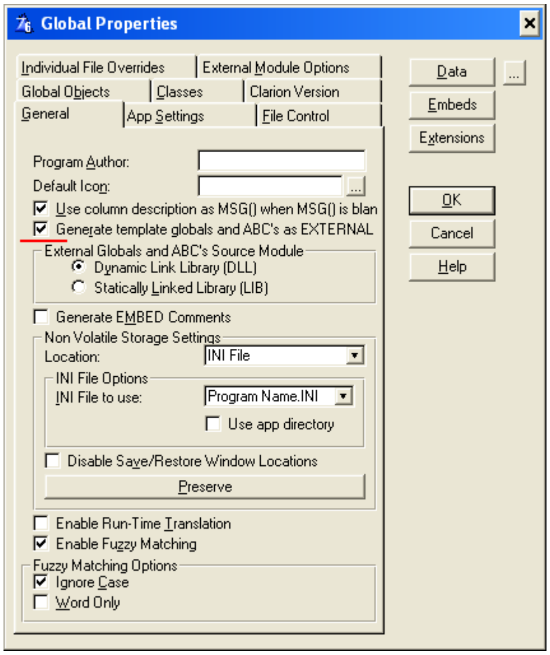
Oh, yes, it helps if I put all the code I intend to test into the global `FileManager` embeds.

In the demo, downloadable at the end of this article, see Data.APP. Data.APP contains nothing but the embedded `FileManager` code. And, the global settings, shown above.

The "other" app

The main app, FM.APP, is the same app as in previous articles with the global `FileManager` embedded code removed.

To make FM.APP work, I added DATA.LIB to the project (Application | Insert Module | External DLL and select the LIB that goes with the data DLL). Then, tell the app that template globals and ABC's are external:



Program Author:

Default Icon:

☑ Use column description as MSG() when MSG() is blan

☑ Generate template globals and ABC's as EXTERNAL

External Globals and ABC's Source Module

● Dynamic Link Library (DLL)

○ Statically Linked Library (LIB)

□ Generate EMBED Comments

Non Volatile Storage Settings

Location: INI File

INI File Options

INI File to use: Program Name.INI

□ Use app directory

□ Disable Save/Restore Window Locations

□ Enable Run-Time Translation

☑ Enable Fuzzy Matching

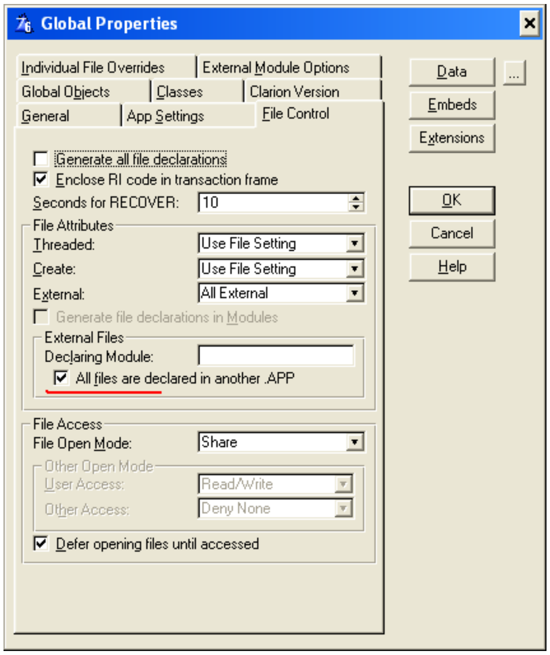
Fuzzy Matching Options

☑ Ignore Case

□ Word Only

Figure 4. Declare ABC's "External"

and confirm that all data is external:



□ Generate all file declarations

☑ Enclose RI code in transaction frame

Seconds for RECOVER: 10

File Attributes

Threaded: Use File Setting

Create: Use File Setting

External: All External

□ Generate file declarations in Modules

External Files

Declaring Module:

☑ All files are declared in another .APP

File Access

File Open Mode: Share

Other Open Mode

User Access: Read/Write

Other Access: Deny None

☑ Defer opening files until accessed

Figure 5. Declare that data is External

Compile and run. Insert a new State. The messages I put in the `FileManager` for the inserts into the `States` file, in the data DLL, appear. Attempt any CRUD operation on `Customers`, my messages appear. Even Browse Customers EIP shows my messages.

Summary

I don’t know why I even questioned whether this would work. I know that `RelationManager`s and `FileManager`s are both created in my apps by the templates. I told the data DLL to compile in all the template globals and ABC classes and objects. By not declaring these "External," I cause them to be exported by the DLL. I told the apps using the data DLL that template globals and ABC’s were declared and exported by another app.

And work, it does. I can embed code in the global ABC objects and that code works in all the apps in the suite. That’s a good thing to know.

Download the source

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

Article comments

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Browse Box Esoterica

By Steven Parker

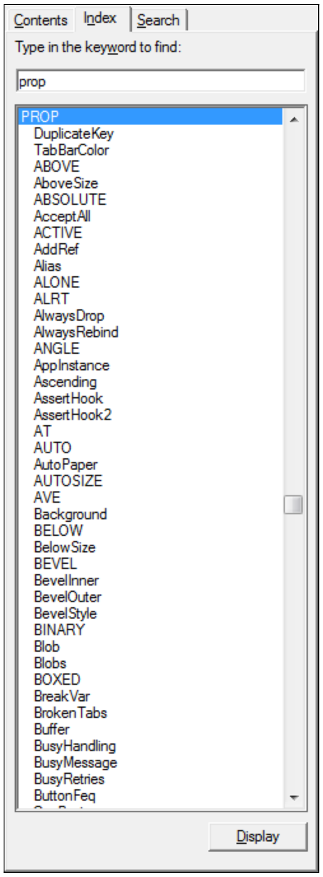
Posted April 20 2010

A number of years ago, when CW1.5 was current, I was presenting some technique or other to my user group. I remember that it was something in a browse box.

What I was demonstrating depended on some embedded code in a browse. The code was what is now called "legacy." At the time, I just used the same code I had used in CDD. I had, after all, established the principle that "Clarion is Clarion." And the browse box was (and remains) a uniquely Clarion innovation. Thus, the code that had worked in Clarion for DOS browse box embeds continued to work in Clarion for Windows. I would expect no less.

"Clarion is Clarion:" Clarion is, first and foremost, a language. Its existing elements change little. Mostly, as it evolves, we get additions to it, extensions. Occasionally something is removed (`Screen`, for example) but "Clarion is (still) Clarion."

One of the attendees, I think it was Carl Barnes, piped up "This is Windows! We use Events now." That wasn't the only change; we also use Properties in Windows. Having read Charles Petzold's Programming Windows, I knew that there were Events all over the place. What I hadn't realized was just how many Properties were now available for me to read, set, forget.

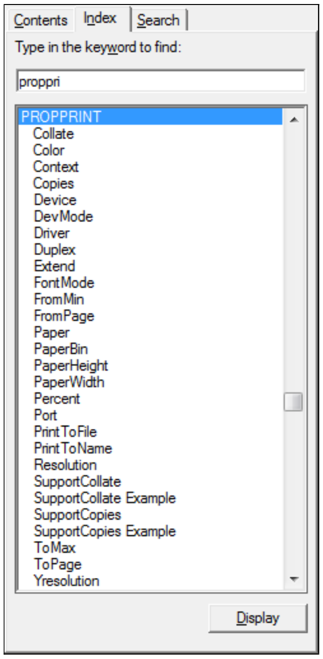


Type in the keyword to find: prop

PROP
DuplicateKey
TabBarColor
ABOVE
AboveSize
ABSOLUTE
AcceptAll
ACTIVE
AddRef
Alias
ALONE
ALRT
AlwaysDrop
AlwaysRebind
ANGLE
AppInstance
Ascending
AssertHook
AssertHook2
AT
AUTO
AutoPaper
AUTOSIZE
AVE
Background
BELOW
BelowSize
BEVEL
BevelInner
BevelOuter
BevelStyle
BINARY
Blob
Blobs
BOXED
BreakVar
BrokenTabs
Buffer
BusyHandling
BusyMessage
BusyRetries
ButtonFeq

Figure 1. Partial list of general windows properties

Figure 1 shows just a small part of the list of window properties – properties of windows – available (this list goes on for another five or six "page down" clicks on the scroll bar). There are additional groups of Properties for specific functional areas:



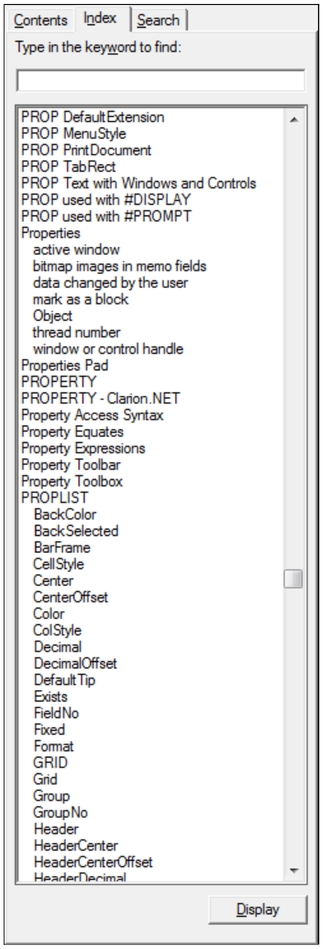
Type in the keyword to find:

proppri

PROPPRINT
Collate
Color
Context
Copies
Device
DevMode
Driver
Duplex
Extend
FontMode
FromMin
FromPage
Paper
PaperBin
PaperHeight
PaperWidth
Percent
Port
PrintToFile
PrintToName
Resolution
SupportCollate
SupportCollate Example
SupportCopies
SupportCopies Example
ToMax
ToPage
Yresolution

Figure 2. Property lists for specialize purposes

Figure 2 shows a list of Properties for printer control.



Type in the keyword to find:

PROP DefaultExtension
PROP MenuStyle
PROP PrintDocument
PROP TabRect
PROP Text with Windows and Controls
PROP used with #DISPLAY
PROP used with #PROMPT
Properties
active window
bitmap images in memo fields
data changed by the user
mark as a block
Object
thread number
window or control handle
Properties Pad
PROPERTY
PROPERTY - Clarion.NET
Property Access Syntax
Property Equates
Property Expressions
Property Toolbar
Property Toolbox
PROPLIST
BackColor
BackSelected
BarFrame
CellStyle
Center
CenterOffset
Color
ColStyle
Decimal
DecimalOffset
DefaultTip
Exists
FieldNo
Fixed
Format
GRID
Grid
Group
GroupNo
Header
HeaderCenter
HeaderCenterOffset
HeaderDecimal

Figure 3. And more Properties

And, the dizzying array of properties goes on and on, as Figure 3 shows.

One very important thing about Properties that is setting a Property takes effect immediately. This code:

```
LOC:ScreenVar = 'Some Text'
```

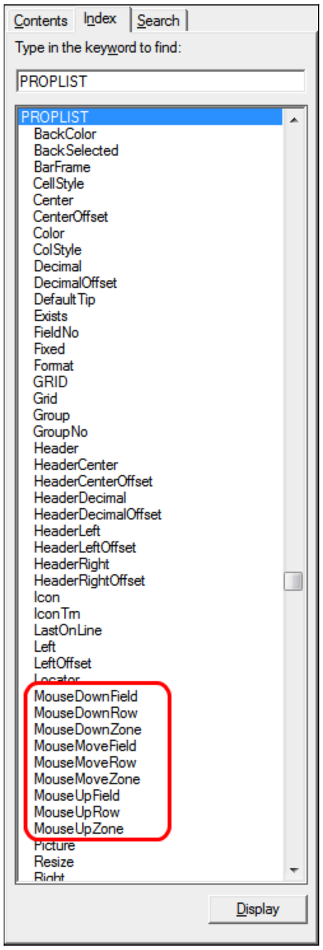
requires a Display or a Refresh or a cycling of the `Accept` loop. But this code:

```
?LOC:ScreenVar{Prop:Text}  = 'Some Text'
```

displays the new value immediately.

What caught my eye in Figure 3 is that there is an entire group of Properties that apply to Lists (`PROPLIST` section).

I thought I knew something about Lists (Browse Boxes) but just look at how many properties are available for them!



Type in the keyword to find:

PROPLIST

PROPLIST
BackColor
BackSelected
BarFrame
CellStyle
Center
CenterOffset
Color
ColStyle
Decimal
DecimalOffset
DefaultTip
Exists
FieldNo
Fixed
Format
GRID
Grid
Group
GroupNo
Header
HeaderCenter
HeaderCenterOffset
HeaderDecimal
HeaderDecimalOffset
HeaderLeft
HeaderLeftOffset
HeaderRight
HeaderRightOffset
Icon
IconTrn
LastOnLine
Left
LeftOffset
Locator
MouseDownField
MouseDownRow
MouseDownZone
MouseMoveField
MouseMoveRow
MouseMoveZone
MouseUpField
MouseUpRow
MouseUpZone
Picture
Resize
Right

Figure 4. List Properties list

I've marked some that look very interesting. If I understand the highlighted Properties correctly, I can use them to determine if a user has clicked on a browse and, if so, where.

I can think of good uses for that kind of information.

Browse box basics

Ignoring, for the moment, the windows controls (the window and the physical list control), a browse is a compendium of three entities:

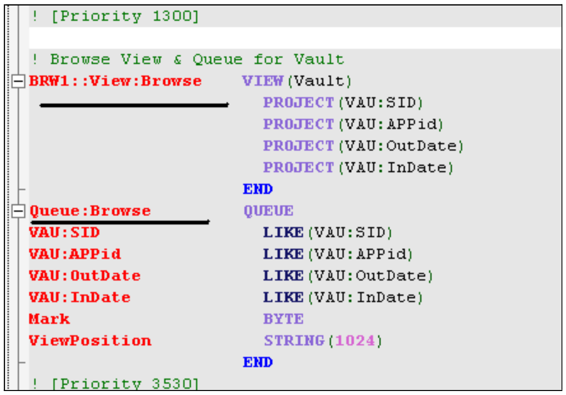
- the file (on disk)
- a view

and

- a queue

When I open a browse procedure, the file on disk is read. It is read into a view. Any filter (and this includes Range limits) is applied to the view (`SetFilter` is a `ViewManager` method, check the online help).

The actual list box, on screen, is filled from the View, indirectly. Specifically, a queue is filled from the `View`.

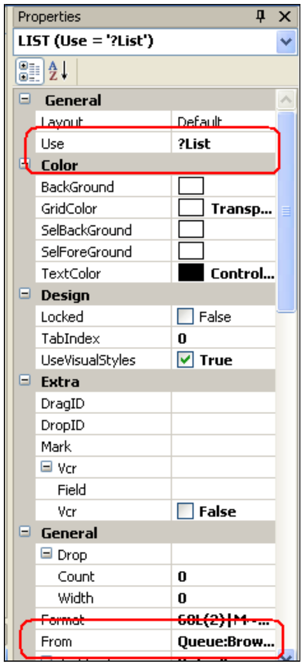


```
! [Priority 1300]

! Browse View & Queue for Vault
BRW1::View:Browse    VIEW(Vault)
                       PROJECT(VAU:SID)
                       PROJECT(VAU:APPid)
                       PROJECT(VAU:OutDate)
                       PROJECT(VAU:InDate)
                     END
Queue:Browse         QUEUE
VAU:SID                LIKE(VAU:SID)
VAU:APPid              LIKE(VAU:APPid)
VAU:OutDate            LIKE(VAU:OutDate)
VAU:InDate             LIKE(VAU:InDate)
Mark                   BYTE
ViewPosition           STRING(1024)
                     END
! [Priority 3530]
```

Figure 5. Generated source showing standard View and Queue

It is the queue that is displayed in the browse box. It is the queue that is the `From` attribute on the browse's properties screen:



Properties

LIST (Use = '?List')

General	
Layout	Default
Use	**?List**
Color	
BackGround	□
GridColor	□ **Transp...**
SelBackGround	□
SelForeGround	□
TextColor	■ **Control...**
Design	
Locked	□ False
TabIndex	**0**
UseVisualStyles	☑ **True**
Extra	
DragID	
DropID	
Mark	
Vcr	
Field	
Vcr	□ **False**
General	
Drop	
Count	**0**
Width	**0**
Format	**60L(2)|M-...**
From	**Queue:Brow...**

Figure 6. Browse properties

Note that the browse box has a `Use` variable. This is what allows referring to the browse box in code (that's what `Use` variables *do*), as well as reading and setting Properties. The `From` property names the queue to display.

The key thing is that, because a queue is used, the entire range of queue commands can be used on it, on need and on demand. (I could, though I cannot imagine why, for example, actually add an element to the browse's queue....)

Getting a specific browse box row

The demo app, built with 7.1.6849, is downloadable at the end of this article. All DLLs necessary to run it are included, as well as the generated CLWs.

Because the list box is (`From`) a queue, I can get the highlighted record with a standard `Get`. To `Get` a queue record, all I need is the queue Label and the pointer (queue element number).

The queue in this procedure is `Queue:Browse` (see Figure 6, above). To get the pointer, `Choice(`*`<ListUseVariable>`*` )` returns what I need (`?List`, in Figure 6).

In the demo app, see the `GetQueueRecord` procedure. From the main menu, select Browse | Get Queue Record and double click on a line in the browse.

In the `AlertKey` embed for the list:

```
Get(Queue:Browse,Choice(?List))
stop('The selected appID is ' & VAU:APPid & |
     ' out date is ' &  Format(VAU:OutDate,@d17))
Return Level:Notify
Select(?List)
```

This code is in the AlertKey embed for the list box (even though I have not ALERTed MouseLeft). Because MouseLeft2 (double click left mouse button) is alerted by default on browses, my Stop message will appear. Because I Return Level:Notify, I do not fall through to the update form. Comment out the Return and the update form *will* appear.

Note that only fields in the View (populated in the list box, or "Hot" and, hence, in the view and therefore in the queue) can be reliably read. If I do not have the field and, for some reason, cannot populate it in the View, I want to read in the View but I do have a unique identifier field(s) available:

```
Access:<File>.Fetch(<uniqueKey>)
Get(Queue:Browse,Choice(?List))
```

retrieves the entire record.

This is pretty much what the templates do when a user double clicks to update a record. It also means that, should I want, I can act on the information retrieved.

The main use for this technique, however, is when I populate a List control without using the template. In this case, I name a queue I have created and populated before the window displays. Because I can execute Get(Queue:Browse,Choice(?List)) when a user presses a button, I can "emulate" standard Insert/Update/Delete actions on my own queue. I can make a queue behave like a template-managed file/table.

Of rows and columns

The List properties, shown in Figure 4, imply that I can detect the row and/or column that a user clicks on. The relevant properties seem to be

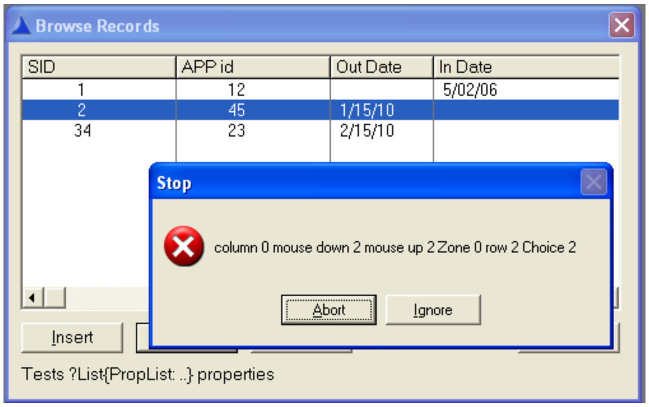
- MouseDownField
- MouseDownRow
- MouseDownZone

and matching properties for "up."

Because MouseLeft2 is already alerted for browse boxes (in order to handle calling the update procedure on mouse double click – this is Standard, and expected, Windows Behavior), I decided to use it to test all these properties in one go:

```
STOP('column ' & ?List{PROP:Column}  & |
  ' mouse down ' &  ?List{PropList:MouseDownField} & |
  ' mouse up ' &  ?List{PropList:MouseUpField} & |
  ' Zone ' & ?List{PROPLIST:MouseDownZone}  & |
  ' row ' & ?List{PROPLIST:MouseDownRow}  & |
  ' Choice ' & CHOICE(?list))
```

In the demo app, from the main menu, select Browse | Test PropList Properties. The procedure is Browse1.



SID	APP id	Out Date	In Date
1	12		5/02/06
2	45	1/15/10	
34	23	2/15/10	

column 0 mouse down 2 mouse up 2 Zone 0 row 2 Choice 2

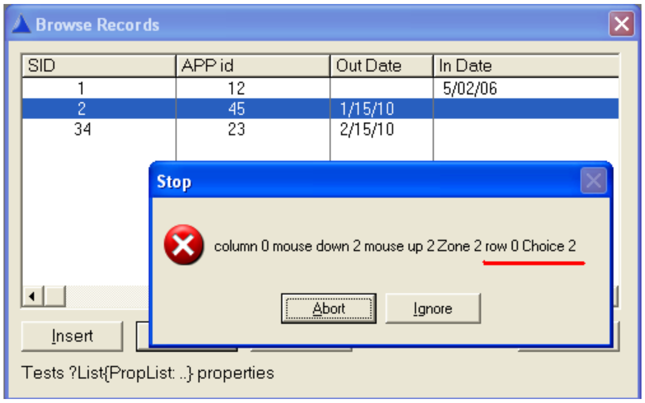
Tests ?List{PropList: ..} properties

Figure 7. Results of double click on second row

Note that Prop:Column returns 0, despite what the online help indicates, and that both PropList:MouseDownField and PropList:MouseUpField return the same value. This value is the ordinal of the column that was under the mouse when I double clicked. Thus, there is nothing to choose between PropList:MouseDownField and PropList:MouseUpField.

Perhaps PropList:MouseDownField and PropList:MouseUpField might return different values if I tested against a single mouse click. This would cover the case where I click down in one place, move my mouse elsewhere and lift there.

Also, PropList:MouseDownRow and Choice(?List) return the same value. This is pretty much as expected. However, if I double click on the browse box header, I get a different result:



SID	APP id	Out Date	In Date
1	12		5/02/06
2	45	1/15/10	
34	23	2/15/10	

column 0 mouse down 2 mouse up 2 Zone 2 row 0 Choice 2

Tests ?List{PropList: ..} properties

Figure 8. Double click on header

Double clicking on the header shows that PropList:MouseDownRow and Choice(?List) are different. Because the second row in the browse was highlighted, in this case, Choice(?List) returns 2. And that is correct. But I clicked on the header and the header (it turns out) is row 0. PropList:MouseDownRow returns the correct value for the header.

In other words, to detect a user click on the header, Choice(?List) is incorrect. To detect a user clicking on the header, check ?List{PropList:MouseDownRow} = 0. But, more important: because Choice(?List) returns a valid value when clicking somewhere other than a Browse Box line, I must henceforth be very careful about how and when I use Choice(?List).

Using column selected

Also note that if you select Browse | Test PropList Properties in the demo app and click "Ignore" on the Stop message, the update form appears.

Suppose I want to offer the user the option of filtering the browse on the column s/he just double clicked? (In a production control app I worked on, this was implemented to allow a user to filter a browse on a single device or production line to see the events for that device or production line.) If the user indicates they do want to filter, I set the filter and return to the browse. If the user indicates they do not want to filter, I drop through to the form:

```
If KeyCode() = MouseLeft2
  NDX =  ?Browse:1{PropList:MouseDownField}
  ! Needed for C6 but not  C7
  Get(Queue:Browse:1,Choice(?Browse:1))
  ! no point if not on a  column I'm interested in
  !Case  ?Browse:1{PropList:MouseUpField}
  ! not testable in C6 but okay in C7
  Case NDX
  Of 3                     ! PRD Line
    Get(Queue:Browse:1,Choice(?Browse:1))
    LOC:FilterVar =  BRW1.Q.TRN:LineCode
    ! Save queue var to a  local
    CASE MESSAGE('Show only  records with Line Code = "' & |
        Clip(LOC:FilterVar)  & '?"',  'Filter?', |
        ICON:Question,|
        BUTTON:Yes+BUTTON:No,  BUTTON:Yes, 0)
    OF BUTTON:Yes
      BRW1.SetFilter('(TRN:LineCode = LOC:FilterVar)')
      ThisWindow.Reset(1)
    OF BUTTON:No
      ! drops through to update form
    END !CASE
  Of 4                       ! device
    !Etc
  End
```

There are several points to note in the code above:

A. In C6, I found that ?Browse:1{PropList:MouseDownField} was not directly testable in my Case structure. While my Stop statements returned the expected column values, as an argument of Case, it did not. Storing it in a local variable solved that problem. In C7, ?Browse:1{PropList:MouseDownField} is directly testable in the Case statement.
B. I am not certain that the first Get(Queue:Browse:1,Choice(?Browse:1) is actually required. On the other hand, it doesn't hurt to be certain.
C. Because I only want to offer the "Do you want to filter?" option on some columns, any column number not in the Case will simply fall through to the update procedure.
D. Similarly, if the user clicks "No" on the message, the code will fall through to the update procedure.
E. When the user does double click on a column for which I want to offer the message, I Get the queue record. I copy the relevant queue variable to a local variable:

```
LOC:FilterVar = BRW1.Q.TRN:LineCode
BRW1.SetFilter('(TRN:LineCode =  LOC:FilterVar)')
```

This allows me to create a SetFilter statement that is readable (i.e., doesn't use the same variable on both sides of the equal sign). I could have avoided the use of a local variable by using the queue variable directly:

```
BRW1.SetFilter('(TRN:LineCode = BRW1.Q.TRN:LineCode)')
```

It's entirely a matter of style and readability.

In the demo app, Browse | Act Only On Certain Columns behaves fairly close to the code, above (without actually setting the filter).

Using the row selected

If all I were interested in was the queue record selected,

```
Get(Queue:Browse,Choice(?List))
```

is sufficient, as discussed above. If I need to be sure I have the entire underlying record from the file, I can follow the Get with a Fetch on the unique key.

Of more interest is that I can tell whether the user clicked on the header or on a row in the list.

Compare Figures 7 and 8, above. In Figure 7, I clicked on the second row of the list.

MouseDownRow returns 2 and "Zone" (MouseDownZone) returns 0. In Figure 8, I clicked on the header. MouseDownZone returned 2, row returned 0 and Choice remained 2.

This means that if Row is 0 or Zone is 2 (Zone might be different in your app, so check), the user clicked on the header. And, supposing I have "Enable Sort Header" on, I can display a string indicating the selected sort order:

```
Case  KeyCode()
Of MouseLeft orof 0001H
  !  Both zone and Row work here
  If ?List{PROPLIST:MouseDownZone} = 2
    !If ?List{PROPLIST:MouseDownRow} = 0
    CASE ?List{PropList:MouseDownField}
    Of 1
      ?String1{Prop:Text} = 'Sorted  by ID'
    Of 2
      ?STRING1{Prop:Text} = 'Sorted  by App'
    Of 3
      ?String1{Prop:Text} = 'Sorted  by Out Date'
    Of 4
      ?String1{Prop:Text} = 'Sorted  by In Date'
    End
  END
End
```

In the demo app, select Browse | Sort Header.

Sort headers

Enabling dynamic sorting by clicking on the header column presents a major problem.

Header sorting is implemented on MouseLeft, a single click of the left mouse button.

For my code, above, to work, I need to Alert(MouseLeft) on the browse box. My code, as shown above, would go in the AlertKey embed. However, and this code *is* in the AlertKey embed in the demo app, the code, as well as a Stop in PreAlertKey, never executes.

Enabling Sort Headers for the browse is designed to act on MouseLeft. So, my AlertKey code is by-passed on the browse box, very much defeating my current purpose. You can confirm that those embeds do get called, on MouseLeft2, by double clicking in the SortHeader procedure. But if you click just once, no Stops appear. Turn off Sort Headers and MouseLeft starts responding again.

So, the combination of Sort Headers and ALERT(MouseLeft) gives the result that any

MouseLeft code in the AlertKey and PreAlertKey embeds is ignored. MouseLeft AlertKey handling has already been handled by the Sort Header code. This means that another method of accomplishing my goal is required.

However, because "Clarion is Clarion" and Clarion features the all knowing Accept loop, that alternate method is readily available. Examining my code, I realize that I *can* safely put it in TakeEvent, the Accept loop. Because my code is limited by two conditions, it will only execute if both

```
 If KeyCode() = MouseLeft
```

and

```
 If ?List{PROPLIST:MouseDownZone} = 2
```

are true. Those two conditions can only be met when the user clicks on the list header to change sorting.

In the demo app, Browse | Sort Header, I left code in AlertKey as well as in TakeEvent. The code in TakeEvent works. When you click on the header, you will see a text field on the window show the selected sort order.

Summary

Setting properties not is not only more efficient (no need to refresh the window), there are many more things I control in code. Examining the documentation on Prop: is well worth the time.

Also, I think I need to take a more systematic look at ALERTing keys and their interactions with my code and with Sort Headers. Perhaps, another article....

Download the source

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

Article comments

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

New Feature: The Clarion Roadmap

By Dave Harms

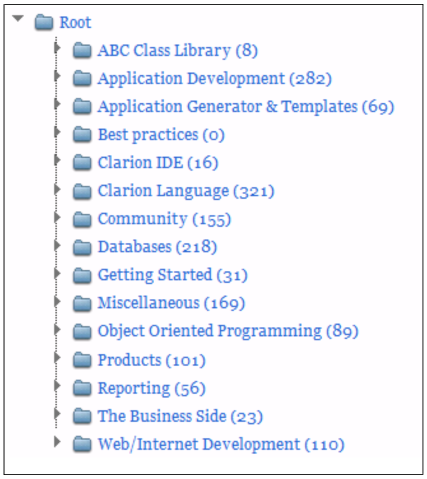
Posted April 21 2010

Clarion Magazine contains a wealth of information, but it hasn't always been easy to get the big picture. Our topical index helped, but it was difficult to maintain and not very flexible.

Our new "roadmap" view of the magazine lets you easily navigate to your areas of interest. Why do we call it a roadmap? Because a roadmap gives you the lay of the land, and helps you reach your destination. It also shows you destinations you may not have considered.

> **Note:** The following screen shots are taken from an admin page, which is also used to create and maintain the roadmap(s); we hope to have the public-facing page up on the beta site early next week). This is a preliminary version of the outline.

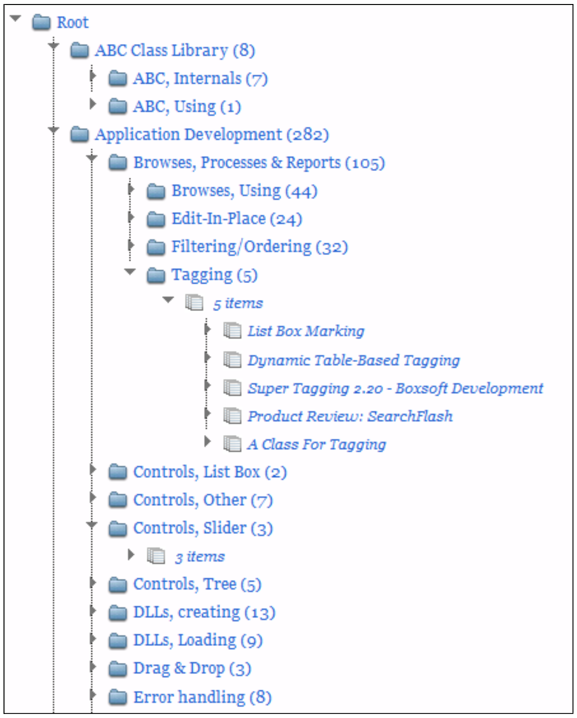
The default ClarionMag roadmap view is an outline, much like a table of contents (Figure 1).



- Root
  - ABC Class Library (8)
  - Application Development (282)
  - Application Generator & Templates (69)
  - Best practices (0)
  - Clarion IDE (16)
  - Clarion Language (321)
  - Community (155)
  - Databases (218)
  - Getting Started (31)
  - Miscellaneous (169)
  - Object Oriented Programming (89)
  - Products (101)
  - Reporting (56)
  - The Business Side (23)
  - Web/Internet Development (110)

Figure 1. The top-level outline view

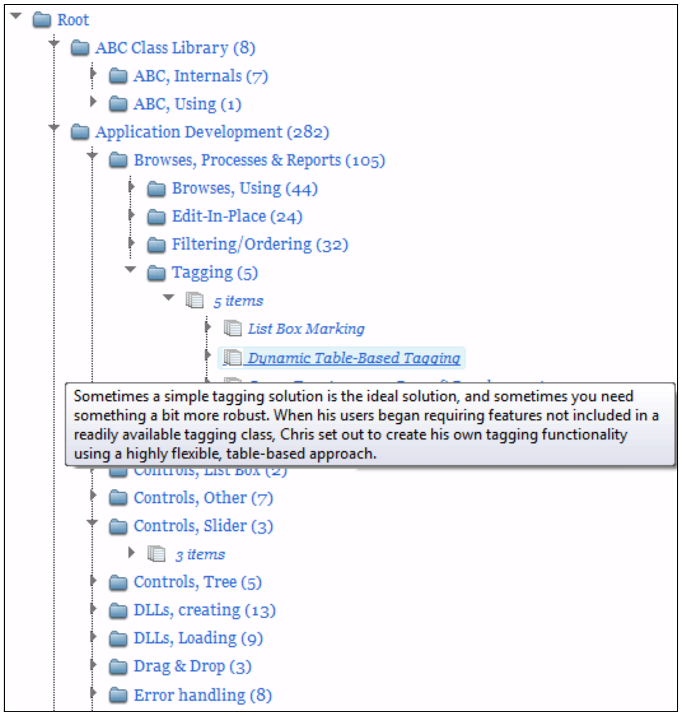
You can click on nodes individually to expand them, or you can use a spinbox to expand all nodes to a specified level. In Figure 2 several nodes have been expanded manually, including the Tagging heading with five articles.



- Root
  - ABC Class Library (8)
    - ABC, Internals (7)
    - ABC, Using (1)
  - Application Development (282)
    - Browses, Processes & Reports (105)
      - Browses, Using (44)
      - Edit-In-Place (24)
      - Filtering/Ordering (32)
      - Tagging (5)
        - *5 items*
          - *List Box Marking*
          - *Dynamic Table-Based Tagging*
          - *Super Tagging 2.20 - Boxsoft Development*
          - *Product Review: SearchFlash*
          - *A Class For Tagging*
    - Controls, List Box (2)
    - Controls, Other (7)
    - Controls, Slider (3)
      - *3 items*
    - Controls, Tree (5)
    - DLLs, creating (13)
    - DLLs, Loading (9)
    - Drag & Drop (3)
    - Error handling (8)

Loading all of the articles under each outline item would take a lot of time and make the page too slow, so instead the articles are loaded on demand, when you expand an articles node. Besides the articles (items) node, look for product, video, audio and web nodes in the near future.

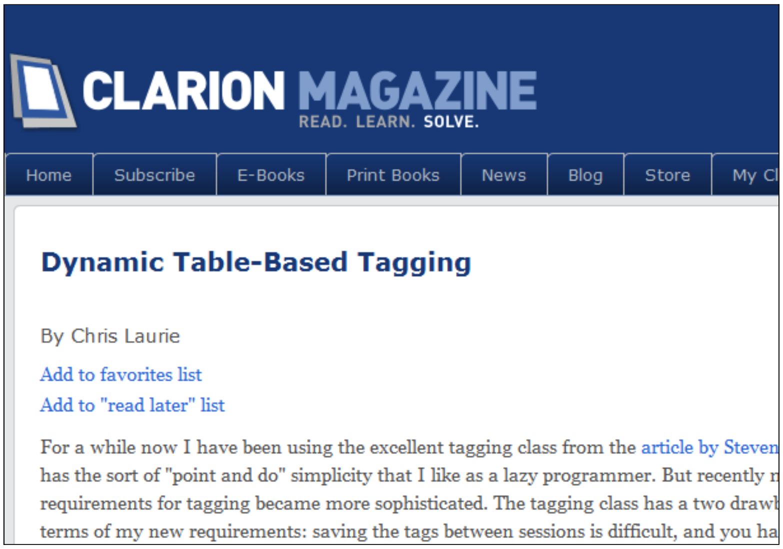
Hover over an article title and you can read the summary (Figure 3).



Sometimes a simple tagging solution is the ideal solution, and sometimes you need something a bit more robust. When his users began requiring features not included in a readily available tagging class, Chris set out to create his own tagging functionality using a highly flexible, table-based approach.

Figure 3. Reading an article summary as a tooltip

Click on the article to view it. At that point you'll have the option to add it to your "favorites" list or your "read later" list (Figure 4).



Dynamic Table-Based Tagging

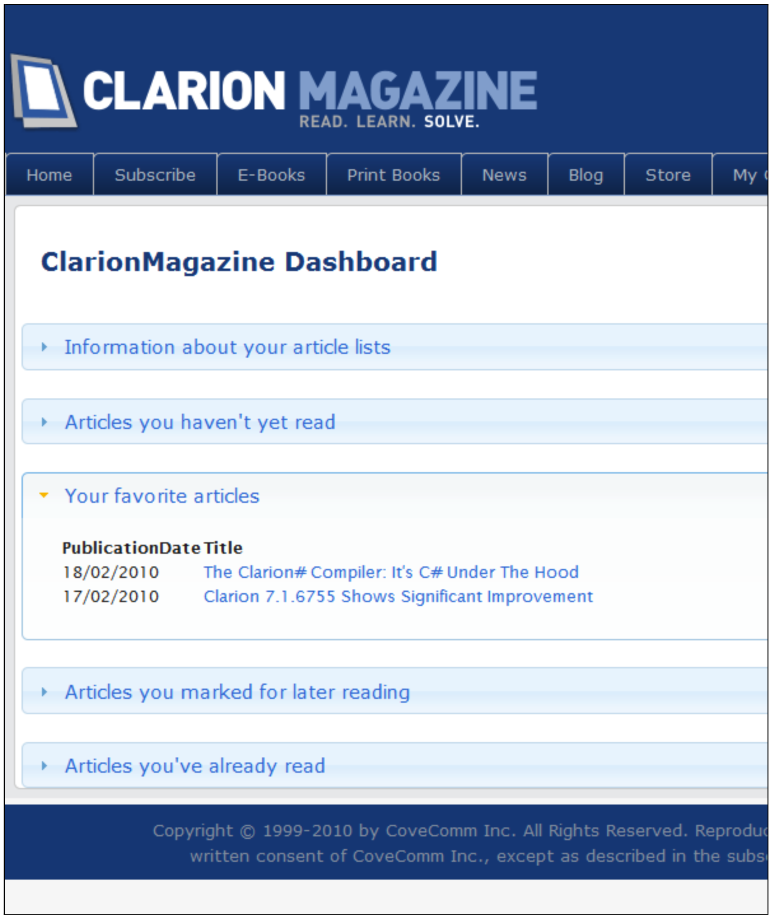
By Chris Laurie

Add to favorites list
Add to "read later" list

For a while now I have been using the excellent tagging class from the article by Steven
has the sort of "point and do" simplicity that I like as a lazy programmer. But recently n
requirements for tagging became more sophisticated. The tagging class has a two drawb
terms of my new requirements: saving the tags between sessions is difficult, and you ha

Figure 4. Links to add the article to your favorites and read later lists

You can view your lists via the My ClarionMag page (Figure 5).



ClarionMagazine Dashboard

Information about your article lists

Articles you haven't yet read

Your favorite articles

PublicationDate	Title
18/02/2010	The Clarion# Compiler: It's C# Under The Hood
17/02/2010	Clarion 7.1.6755 Shows Significant Improvement

Articles you marked for later reading

Articles you've already read

Copyright © 1999-2010 by CoveComm Inc. All Rights Reserved. Reproduc
written consent of CoveComm Inc., except as described in the subs

Figure 5. Viewing the article lists.

You can use the article lists independently of the roadmap.

Multiple roadmap views

Sometimes you want to see the whole roadmap, and sometimes you just want to see part of the roadmap. So we've added the ability to filter the roadmap. We'll be providing a number of standard filters (Figure 6 shows the first two, Getting Started and Browses & Forms). You can choose the filter you want from the drop-down list.

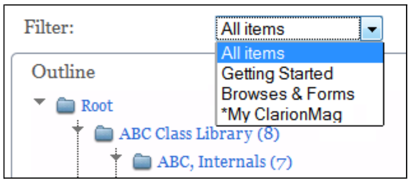

Filter: All items

Figure 6. Choosing a roadmap filter

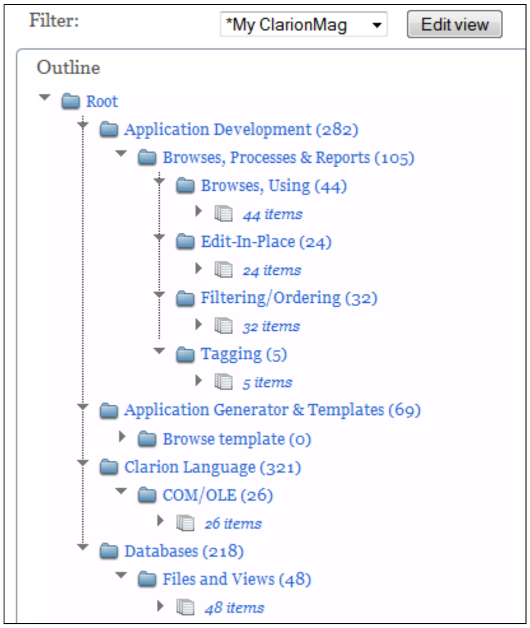


Figure 7. The Getting Started view (under construction)

And if none of the pre-packaged filters suit your taste, you can create your own personalized roadmap filter. Just choose the *My ClarionMag option (name subject to change) from the dropdown list, and an Edit button appears.

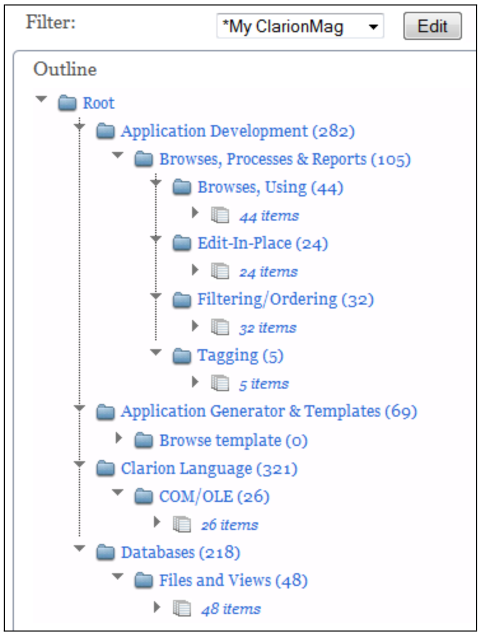
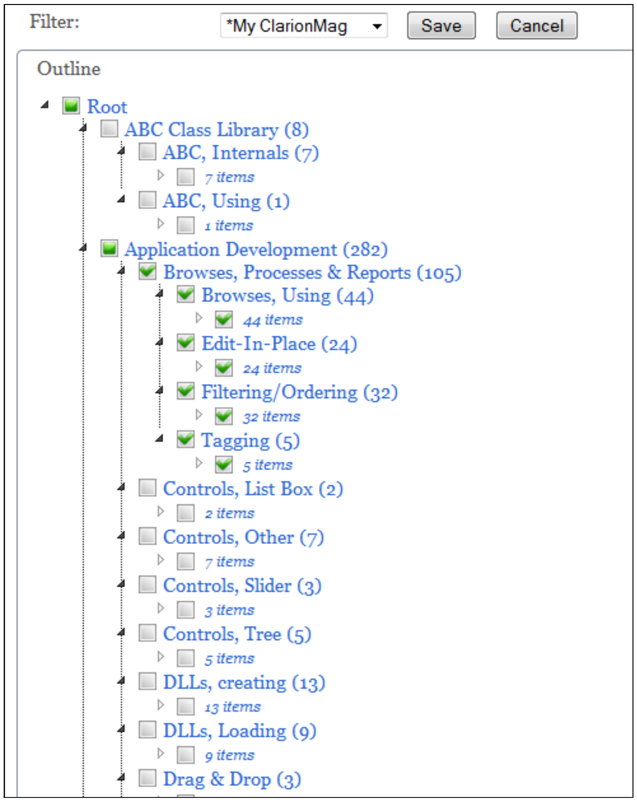


Figure 8. The My ClarionMag filter

Click on the Edit button and you see a view of the roadmap with checkboxes by each topic.



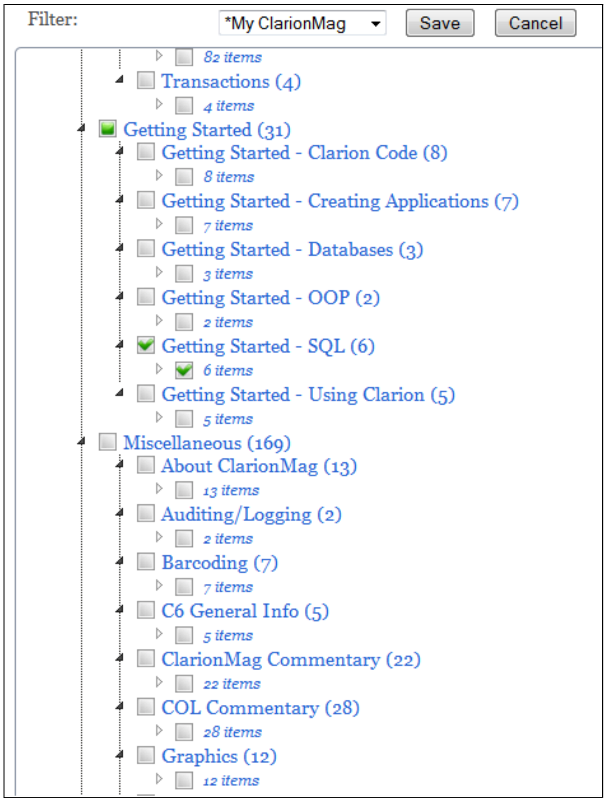
Filter: *My ClarionMag ▾ [Save] [Cancel]

Outline

- ▣ Root
  - ☐ ABC Class Library (8)
    - ☐ ABC, Internals (7)
      - ☐ *7 items*
    - ☐ ABC, Using (1)
      - ☐ *1 items*
  - ▣ Application Development (282)
    - ☑ Browses, Processes & Reports (105)
      - ☑ Browses, Using (44)
        - ☑ *44 items*
      - ☑ Edit-In-Place (24)
        - ☑ *24 items*
      - ☑ Filtering/Ordering (32)
        - ☑ *32 items*
      - ☑ Tagging (5)
        - ☑ *5 items*
    - ☐ Controls, List Box (2)
      - ☐ *2 items*
    - ☐ Controls, Other (7)
      - ☐ *7 items*
    - ☐ Controls, Slider (3)
      - ☐ *3 items*
    - ☐ Controls, Tree (5)
      - ☐ *5 items*
    - ☐ DLLs, creating (13)
      - ☐ *13 items*
    - ☐ DLLs, Loading (9)
      - ☐ *9 items*
    - ☐ Drag & Drop (3)

Figure 9. Creating the filter

Select the topics you're interested in. Click Save, and you now have your very own personalized roadmap filter. You can always go back and make changes. In Figure 10 I've scrolled down and added the Getting Started - SQL topic.

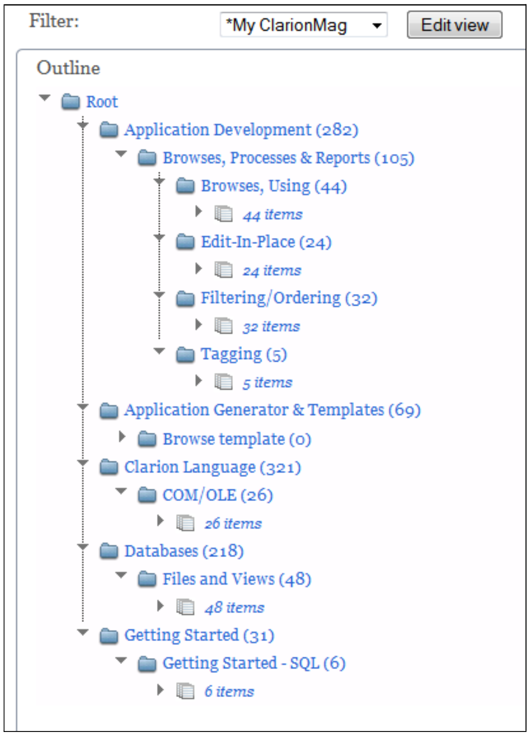


Filter: *My ClarionMag ▾ Save Cancel

- ☐ *82 items*
- ☐ Transactions (4)
  - ☐ *4 items*
- ☑ Getting Started (31)
  - ☐ Getting Started - Clarion Code (8)
    - ☐ *8 items*
  - ☐ Getting Started - Creating Applications (7)
    - ☐ *7 items*
  - ☐ Getting Started - Databases (3)
    - ☐ *3 items*
  - ☐ Getting Started - OOP (2)
    - ☐ *2 items*
  - ☑ Getting Started - SQL (6)
    - ☑ *6 items*
  - ☐ Getting Started - Using Clarion (5)
    - ☐ *5 items*
- ☐ Miscellaneous (169)
  - ☐ About ClarionMag (13)
    - ☐ *13 items*
  - ☐ Auditing/Logging (2)
    - ☐ *2 items*
  - ☐ Barcoding (7)
    - ☐ *7 items*
  - ☐ C6 General Info (5)
    - ☐ *5 items*
  - ☐ ClarionMag Commentary (22)
    - ☐ *22 items*
  - ☐ COL Commentary (28)
    - ☐ *28 items*
  - ☐ Graphics (12)
    - ☐ *12 items*

Figure 10. Adding another topic to the filter

Figure 11 shows the updated outline view with my personal filter applied.



Filter: *My ClarionMag ▾ Edit view

Outline

- Root
  - Application Development (282)
    - Browses, Processes & Reports (105)
      - Browses, Using (44)
        - *44 items*
      - Edit-In-Place (24)
        - *24 items*
      - Filtering/Ordering (32)
        - *32 items*
      - Tagging (5)
        - *5 items*
  - Application Generator & Templates (69)
    - Browse template (0)
  - Clarion Language (321)
    - COM/OLE (26)
      - *26 items*
  - Databases (218)
    - Files and Views (48)
      - *48 items*
  - Getting Started (31)
    - Getting Started - SQL (6)
      - *6 items*

Figure 11. The updated filter has been applied

As new items added to Clarion Magazine and tagged, they're automatically added to the roadmap via their tags and they appear in your personalized roadmap view where appropriate.

Future roadmap enhancements

Currently only articles are listed in the outline view, but we'll be adding third party products, blogs, podcasts, screencasts and other resources to the roadmap.

For .NET

We will have a Clarion#-specific roadmap view as well. For the larger world of .NET development we'll be applying this same roadmap technology in our subscription-based DevRoadmaps.com site.

David Harms is an independent software developer and the editor and publisher of Clarion Magazine. He is also co-author with Ross Santos of Developing Clarion for Windows Applications, published by SAMS (1995), and has written or co-written several Java books. David is a member of the American Society of Journalists and Authors (ASJA).

Article comments

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Alert(MouseLeft)

By Steven Parker

Posted April 21 2010

In Browse Box Esoterica, I explored several interesting properties of list controls. The list box properties I found most useful were (assuming the Use variable of my browse is "`?List`"):

- `Choice(?List)`: Returns the highlighted list box row, regardless of where the user clicks
- `?List{PROPLIST:MouseDownRow}`: Returns the list box row on which the user clicked; a return value of 0 indicates the user clicked on the list box header
- `?List{PROPLIST:MouseDownZone}`: Returns different values if the user clicks on the list box header or on the list itself (in my sample app, 2 for the header, 0 for the list)
- `?List{PropList:MouseDownField}`: This one is the most interesting; it tells me which list box column the user clicked....

If I want to distinguish a click on the list box header from a click on its body, I seem to have a choice between `PROPLIST:MouseDownZone` and `?List{PROPLIST:MouseDownRow} = 0`. Checking what SoftVelocity itself does (for example, implementing the Sort Headers feature) reveals that they use `?List{PROPLIST:MouseDownRow} = 0`. Since I am not certain that `?List{PROPLIST:MouseDownZone}` always returns 0 and 2, as it did in my tests, but SoftVelocity has hard coded `?List{PROPLIST:MouseDownRow} = 0` in the templates and ABC classes, I see no reason to use anything else.

In that previous article, I encountered problems. I found that when I `Alert`ed `MouseLeft` on a list and enabled Sort Headers for the list, `MouseLeft` code in the `PreAlertKey` and `AlertKey` embeds was no longer called.

This needed more rigorous testing than I gave it before. I had been inserting a lot of code for testing and I think I may have stepped on my own results.

People.APP

I started with the infamous People.APP as distributed with Clarion 7.1.6849. I copied the original Browsepeople procedure and systematically modified it by adding `STOP` statements, ALERTing keys and enabling Sort Headers.

Here are the results, procedure by procedure:

Browsepeople: This is the standard procedure, as supplied by SoftVelocity. There is no embedded code. There are no ALERTed keys. Sort Headers was enabled but I turned it off for the moment.

There is absolutely nothing remarkable when running the procedure (from the Menu bar: Browse | Browse the people file). Click on records, scroll up and down, page up or down, everything works exactly as one would expect.

BrowsepeopleSTOPS: In this copy of the original procedure, STOP statements are added to the PreAlertKey and AlertKey embeds.

Again, there is little remarkable found running this procedure (from the Menu bar: Browse | Browse with "Alert Stops"). Click on records, scroll up and down, page up or down, everything works exactly as one would expect.

None of the STOPs appears. This, of course, is correct as no keys have been ALERTed for the browse box.

BrowsepeopleSTOPSandF2: To prove that ALERTed keys are, indeed, handled correctly, this copy of the original features the Pre- and AlertKey STOP statements and the F2 key is ALERTed on the browse box. Sort Headers are not enabled.

From the Menu bar: Browse | Browse with "Alert Stops" and Alert(F2) runs the procedure. It behaves exactly as expected on clicks, scroll events, etc. When the list box is selected, however, and you press the F2 key, you will see the two STOP messages.

ALERT keys are working. QED.

BrowsepeopleSTOPSandSortHeader: For this iteration, from the Menu bar, select Browse | Browse with Stops and Sort Header, I left the STOP statements in place and enabled Sort Headers. I left F2 ALERTed for the procedure.

Absolutely nothing unusual happens on running this procedure. Clicking, scrolling, clicking on the browse header all behave as expected. When F2 is pressed, the messages appear. In short, everything behaves normally.

This conclusively demonstrates that when I enabled Sort Headers for the list box, code in the PreAlertKey and AlertKey embeds is called for a key other than MouseLeft.

Sort Headers

It is, however, important to note a small bit of template code added to the ACCEPT loop when Sort Headers are enabled. At ThisWindow.TakeEvent, Before Parent Call but after Top of CYCLE/Break support (around Priority 3000), Sort Headers inserts:

```
IF BRW1::SortHeader.TakeEvents()
  RETURN  Level:Notify
END
```

Word to the wise: if Sort Headers is enabled and you need to put code in the ACCEPT loop, do it at Priority 3000 or earlier. If not, it will not execute when the user clicks on a browse box header.

Was this the cause of code in the PreAlertKey and AlertKey embeds failing to be called for ALERT(MouseLeft)? Or, had I made some mistake in my sample application?

BrowsepeopleSTOPSandMouseLeft: In this iteration (Browse | Browse People with STOPs and Alert(MouseLeft) to run the procedure), I ALERTed MouseLeft on the browse box. Sort Headers is off.

Clicking on a record in the list immediately results in both of my STOPs, PreAlertKey and AlertKey.

What is wrong is that the record I just clicked on is not selected. The highlight bar does not move to the line I was pretty sure I had selected. This is just not acceptable program behavior.

MouseLeft is referred to in a number of the template/ABC files. BrowseRoutineAlertKey, a Group in CTLBROW.TPW refers to MouseLeft. There are several instance in BRWEXT.CLW, where there appears to be handling for AlertKey events. cBrowse.OnAlertKey also caught my eye.

But, trying to debug through template and ABC class code is not a task I wish to undertake. I *have* satisfied myself that there is something special about Alert(MouseLeft). If I want to do something special on MouseLeft, formatting and displaying a string with the selected sort order for example, ALERT(MouseLeft) is actually going to get in my way.

This means that ALERT(MouseLeft) may not be such a good idea, at least on list boxes.

What *was* the problem?

Why, in the previous demo app, did the STOPs in PreAlertKey and AlertKey not display when expected? They appear with I ALERT(F2) and when I ALERT(MouseLeft). When I enable Sort Headers and ALERT(F2), they appear.

However, with Sort Headers enabled and ALERT(MouseLeft), they will not appear. Perhaps because of the IF BRW1::SortHeader.TakeEvents() code (shown above) inserted by the templates on enabling Sort Headers, the code does not fall through to the AlertKey embeds. A left mouse click is, after all, a Sort Header event (or it is *the* Sort Header event).

Satisfy yourself; in the demo app selected Browse | Browse with Sort Headers and MouseLeft. The procedure is BrowsepeopleSTOPSandSortHeaderAndMouse. It has the

STOPs in the AlertKey embeds, Sort Headers enabled and ALERT(MouseLeft) on the browse box. In other words, it is the BrowsepeopleSTOPSandSortHeader procedure plus ALERT(MouseLeft). The STOPs that previously appeared no longer will. Clearly, ALERT (MouseLeft) does not get along with enabling Sort Headers.

Clarion remains "Clarion"

Clarion is Clarion and Clarion features the all knowing Accept loop. In days of DOS, Accept was described as the navigation manager. Accept knows what key was pressed and, if there is a default behavior for that key (e.g., down arrow when on a list box or tab key, anywhere), ensures that the default behavior happens.

So, the window's Accept loop (found at ThisWindow.TakeEvent) knows if I press MouseLeft (ALERTed or not). I have already determined that I can check if a row of the browse has been clicked (?Browse:1{PROPLIST:MouseDownRow} > 0, as discussed above).

So,

```
If  KEYCODE() =  MouseLeft
  If ?Browse:1{PROPLIST:MouseDownRow} > 0  !  not the header
```

should trigger only when I single click on the list box.

In the demo app, Browse | BrowsePeopleTakeEvent includes this code in ThisWindow.TakeEvent:

```
If  KEYCODE() =  MouseLeft
  Stop('mouseleft pressed')
  If  Selected() = ?Browse:1
    Stop('list selected on ?Browse:1')
  End
  If ?Browse:1{PROPLIST:MouseDownRow} > 0  !  not the header
    Stop('List Selected on MouseDownRow')
  End
End
```

If you run this procedure, you will find that the STOPs come up as expected. And you will see that the selector bar moves to the record clicked. (There are other solutions to the problem of the bar not moving to the clicked-on record; but they are seriously messy.)

You will also note that the If Selected() = ?Browse:1 STOP code executes. Puzzling. But, following SoftVelocity's lead, If ?Browse:1{PROPLIST:MouseDownRow} > 0 is known to be reliable.

Much better. The clicked on row is highlighting.

However (isn't that always the way it is?), if you click one of the buttons, Insert or Change or ..., you'll get the messages. You will not get the appropriate action. Or, you will not get it unless you click the button again (and, on exiting the update form, you will get the `Selected() = ?Browse:1 STOP` that didn't execute before).

The procedure will work as expected if the `STOP`s are commented out. This proves Russ Eggen's long time contention about `STOP`s and `MESSAGE`s changing the way windows behave.

And, that may be sufficient.

If, however, I get a bit paranoid, I can check the field clicked on and act only when it is the browse box:

```
If  KEYCODE() =  MouseLeft
  Case  Field()
  Of ?Insert:2
  orof ?Change:2
  orof ?Delete:2
  orof ?Close
  orof ?Print
  ELSE
    Stop('mouseleft pressed')
    If  Selected() = ?Browse:1
      Stop('list selected on ?Browse:1')
    End
    If ?Browse:1{PROPLIST:MouseDownRow}  > 0   ! not the header
      Stop('List Selected on MouseDownRow')
    End
  End
End
```

Alternately:

```
If  KEYCODE() =  MouseLeft and Field() =  ?Browse:1
  Stop('mouseleft pressed')
  If  Selected() = ?Browse:1
    Stop('list selected on ?Browse:1')
  End
  If ?Browse:1{PROPLIST:MouseDownRow} > 0  !  not the header
    Stop('List Selected on MouseDownRow')
  End
End
```

also works.

Summary

So, Clarion *is* Clarion. MouseLeft is handled unlike other `ALERT`ed keys (in combination with Sort Headers, `ALERT`ing `MouseLeft` still has to be handled outside of the browse's `AlertKey` embed – so, don't use both in a single procedure!). But because Clarion is Clarion, the difference is not insurmountable. The all knowing `Accept` loop is available for me to do things the way I want them done.

Yet, I cannot resist saying that Clarion *is* Clarion, unless it's Clarion dot Net. The all knowing `Accept` loop is not, so far as I know, a dot Net feature.

Download the source

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

Article comments

by Dave Harms on June 10 2010 (comment link)

Correct - there is no analogue to `Accept` in .NET.

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

Why Unicode Matters To You (And SoftVelocity)

By Phillip Carroll

Posted April 30 2010

In Part 1 I covered the history and terminology of Unicode. Now it's time to consider whether relying exclusively on a 15-year-old obsolete Unicode standard (the one established in Windows 95) threatens Clarion with a similar obsolescence.

Microsoft strongly recommends all new programs should support Unicode. Without Unicode support, this is a practical impossibility for Clarion users. From MSDN:

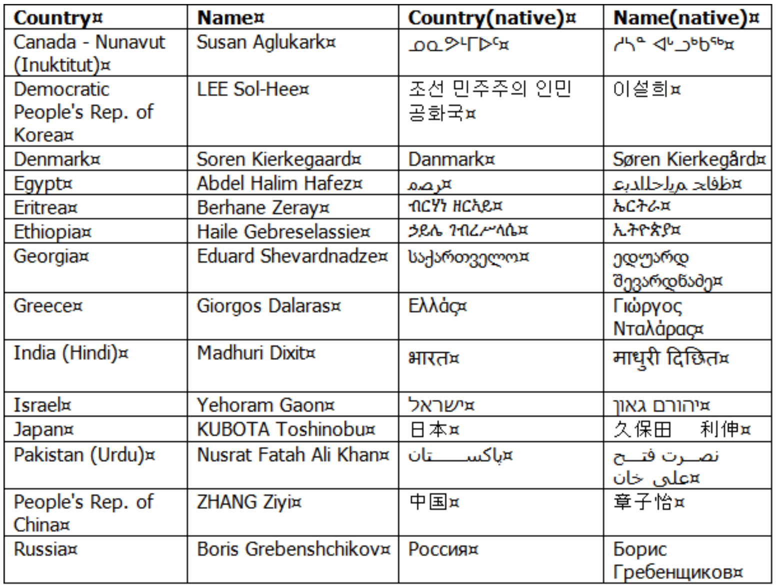
> "New Windows applications should use Unicode to avoid the inconsistencies of varied code pages and for ease of localization."

Without Unicode, multilingual programs are impossible, putting Clarion at a disadvantage with respect to global commerce.

Although as noted above, Unicode is controversial in parts of Asia, Unicode is almost universally adopted in locations where Clarion is currently used or has any practical likelihood of being used.

There is no convincing reason *not* to support Unicode. There is no mystery to Unicode support. The 32-bit WinAPI supports Unicode with a rich variety of functions, and the native Windows internal representation of all text is Unicode. It takes no more work to translate a UTF-8 external format string to the internal UTF-16 format than it takes to translate text encoded with an OEM code page, and vice versa.

Although text encoded in UTF-8 is easily imported into strings in a Clarion program, the following table of UTF-8 encoded values cannot presently be displayed in a Clarion `LIST`, or even on the same window using separate `STRING` or `TEXT` controls for each table cell (Figure 1).



Country¤	**Name¤**	**Country(native)¤**	**Name(native)¤**
Canada - Nunavut (Inuktitut)¤	Susan Aglukark¤	ᓄᓇᕗᒻᒥᐅᑦ¤	ᓱᓴᓐ ᐊᒡᓗᒃᑲᖅ¤
Democratic People's Rep. of Korea¤	LEE Sol-Hee¤	조선 민주주의 인민 공화국¤	이설희¤
Denmark¤	Soren Kierkegaard¤	Danmark¤	Søren Kierkegård¤
Egypt¤	Abdel Halim Hafez¤	مصر¤	عبد الحليم حافظ¤
Eritrea¤	Berhane Zeray¤	ብርሃነ ዘርኣይ¤	ኤርትራ¤
Ethiopia¤	Haile Gebreselassie¤	ኃይሌ ገብረሥላሴ¤	ኢትዮጵያ¤
Georgia¤	Eduard Shevardnadze¤	საქართველო¤	ედუარდ შევარდნაძე¤
Greece¤	Giorgos Dalaras¤	Ελλάς¤	Γιώργος Νταλάρας¤
India (Hindi)¤	Madhuri Dixit¤	भारत¤	माधुरी दिछित¤
Israel¤	Yehoram Gaon¤	ישראל¤	יהורם גאון¤
Japan¤	KUBOTA Toshinobu¤	日本¤	久保田　利伸¤
Pakistan (Urdu)¤	Nusrat Fatah Ali Khan¤	پاکستان¤	نصرت فتح علی خان¤
People's Rep. of China¤	ZHANG Ziyi¤	中国¤	章子怡¤
Russia¤	Boris Grebenshchikov¤	Россия¤	Борис Гребенщиков¤

Figure 1. UTF-8 encoded values

Points to consider:

- The text above was originally pasted into an ASCII file in the editor supplied with the Clarion IDE. All characters immediately appeared with the correct graphics. (It was converted into an image for this article)
- All ASCII text in the plain text file remained exactly as it was, because ASCII is already UTF-8 encoded!
- When saved, the editor popped up a message that the file which was previously in Windows Western format now contained character outside that format, and offered to save the file as UTF-8 format, which was accepted.
- The reader of this may or may not be aware that the editor displaying this text is not concerned with "character sets" or "code pages" or anything of that ilk. The only character-oriented setting is the font. Tahoma was chosen because of its broad language support. (The table text has been converted to a screen shot to avoid any potential encoding problems.)
- By using Unicode, as exemplified above, it doesn't matter whether the text is Greek, English, Hindi, Russian, Georgian, Scandinavian, Hebrew, Arabic, Japanese, Chinese, English, or various other Latin-alphabet languages, or any of dozens of other symbol sets, including mathematical and various other kinds of special symbols or a combination of all of the above. It all can co-exist in the same Unicode UTF-8 encoding. (Of course, one should probably not mix a right-to-left script such as Arabic with western left-to-right scripts in the same string)
- See the accompanying program source file which contains the plain text in an `OMIT` block.

How should Unicode be implemented?

As noted above, the editor supplied with the Clarion 7 IDE accepts UTF-8 natively and recognizes the full range of UTF-8 encoding.

Because UTF-8 contains 7-bit ASCII as a proper subset, essentially all Clarion source files are already in UTF-8 format. Furthermore, if non-ASCII UTF-8 characters are entered into an existing ASCII or extended ASCII source file, the format is automatically recognized whether added using an external editor or using the IDE editor. Any pre-existing "extended ASCII" (ANSI) characters in the file are automatically re-encoded in UTF-8 format with the Unicode code point that has equivalent semantics. For instance the code 80h (the Euro sign in Windows Western coding) is automatically changed to U+20AC (the Unicode Euro sign code point, encoded in UTF-8 as [F2 8A AC]). When the file is saved, the user is warned, and asked whether to save the file in UTF-8 format.

All Clarion string types have the ability to contain the full range of UTF-8 encoded Unicode, because the format consists entirely of 8-bit bytes.

By experimentation (see the accompanying code files) it was quickly observed that CLW source files in UTF-8 format are readily accepted by the compiler. Source files containing UTF-8 in comments, within `OMIT` blocks, and in declared `STRING` constants are compiled, and the program runs as desired, except for the visible output.

Consequently, adopting the UTF-8 format as the standard Clarion 7 source format is practically free.

Because UTF-16 consists of 16 bit values, all characters are byte-reversed on little-endian computers. Therefore, it becomes a crucial matter to know whether a UTF-16 file is in little endian or big endian format. To solve this problem, the code point U+FEFF has been given two meanings. Its primary meaning is "Zero width no break space", which is a typesetting instruction, not an actual character. Secondarily, it is used as a "Byte Order Mark" (BOM). This takes advantage of the fact that U+FFFE is an invalid code point that will never be assigned. Furthermore, neither FF FE nor FE FF is a legal sequence in UTF-8. Therefore, if a text file begins with either FFFE or FEFF it is recognized as a UTF-16 BOM, and the order of the bytes signals whether the file was written in big-endian or little-endian format.

Obviously, UTF-8 being a byte-oriented encoding, the bytes always appear in the same order. Therefore, there is no need for a "byte order mark". Nevertheless, many editors insert the UTF-8 encoding of U+FEFF (EF BB BF) at the beginning of UTF-8 encoded files as a "signature" indicating the file contains UTF-8 as opposed to some ANSI variant. Although the standard permits this mark when necessary, it now prohibits the mark when the encoding format can be determined by other means. (Such as, perhaps, the filename extension)

Recommendations

- Adopt UTF-8 as the standard external format of Unicode in Clarion 7. This would pertain to all files that contain Clarion language source, such as: INC, CLW, TPL, TPW, TXA, TXD and so forth. The Clarion 7 editor already supports it.
- Limit Unicode support to Clarion 7 and later. The Clarion editor in previous IDEs does not support Unicode in any form.
- Provide an application setting in the Clarion 7 IDE to indicate the program is a Unicode program. If this setting is `FALSE`, but a source is included that contains multibyte Unicode characters, the IDE should automatically change the setting to TRUE.
- If the Unicode setting is `FALSE`, the IDE and the runtime should behave as at present.
- Support UTF-8 representation in all Clarion string types. With UTF-8 accepted as a standard representation format for text in Clarion programs, a new Clarion-language Unicode-specific data type is not required. Existing Clarion string types can already store and manipulate UTF-8 strings. While a UTF-8 specific type may offer some minor advantage, the author finds no compelling reason to invent one. Use of existing string types is cost free.
- It is necessary to support strings in Unicode programs that contain some encoding other than UTF-8. Therefore, in a Unicode application:
  - The `CHARSET` property of a control's `FONT` specification should be used by the runtime to determine the translation needed for Windows
  - In the absence of a `CHARSET` in the `FONT` property of a control, a string assigned to the text attribute of the control should be treated as UTF-8
  - All strings assigned to system text properties (such as a `WINDOW` title) should be UTF-8 encoded, and treated as such by the runtime
  - The `CHARSET` property should be ignored by the compiler if used on the `WINDOW FONT` property of a Unicode application
- Although the Clarion 7 editor will automatically recognize when a file contains UTF-8, when it saves a file containing UTF-8 it inserts the so-called "UTF-8 signature" mentioned earlier as the first three characters of the file. If the compiler has been told the application is Unicode, the file will be known to be UTF-8 format. As was stated earlier, the editor recognizes UTF-8 without the mark, and therefore the signature is not only unnecessary, but simultaneously problematic for the Clarion compiler. The compiler will not accept a file containing the mark, and it is almost certain other Clarion programs will also not accept it. The UF-8 specification tolerates this mark in some situations but calls for this mark to be absent when UTF-8 is the *expected* encoding for a particular file type. The editor must be changed to eliminate this mark.
- At least initially, restrict non-ASCII UTF-8 codes in source files to string constants, comments, and `OMIT` blocks. The compiler already accepts UTF-8 in these constructs. There is no particular requirement to change the definition of what is acceptable in

Clarion labels, table names, field names, etc.. This is cost free.

- Do not provide any explicit UTF-8 support for database string fields used in keys or indexes. IOW, treat any strings in key fields as ANSI and use existing collating sequence support via the environment file. UTF-8 exclusion is cost free.
- All strings sourced from `ENTRY` or `TEXT` controls should be stored in the application in UTF-8 format, unless a `CHARSET` is given. This includes any text pasted into the control from the clipboard or entered via keyboards using various locales.
- Support UTF-8 anywhere a string expression is allowed in the language, including the following (non exhaustive):
    - `STRING` and `TEXT` controls
    - `LIST` source variables, including queue fields
    - `BUTTON` text
    - Window titles
    - Tooltips
    - Status text
    - etc.
- Whereas strings containing UTF-8 may require more bytes than characters, string picture tokens are intrinsically character-oriented and are therefore unaffected by UTF-8. A string of 80 characters is a string of 80 characters. This requires no change to the language.

Built-in string manipulation functions:

- Some existing string functions can be used liberally with UTF-8. The reason for this is that UTF-8 was specifically designed with existing string function libraries in mind. For example: Given a string A containing only valid UTF-8 characters, and a string B containing only valid UTF-8 characters, then INSTRING(A,B) will never return anything but a correct answer *with respect to byte position*. The programmer only needs to be aware that one byte does not equal one character.

For example, suppose you want a function to replace character y in a string x with character z.

```
A = 'Whatever € 350'
B = '€'  !Euro Sign in Unicode occupies 3 bytes
C = '$'
B = Replace(A,B,C)
```

Correct way to write replace :

```
Replace  PROCEDURE(x,y,z)
k   LONG
  CODE
  k = INSTRING(y,x)
  x = x[1 : k-1] & z & x[k+LEN(y) : LEN(x)]
  RETURN x
```

Incorrect way to write replace:

```
Replace  PROCEDURE(x,y,z)
k   LONG
  CODE

  k = INSTRING(y,x)
  x[k : k] = z
  RETURN x
```

The author "knew" that x is required to be a single character, and replacing a single character requires only simple indexing. Wrong. With UTF-8, *every character must be assumed to be a multibyte string.* Although `INSTRING` will always return a correct answer with UTF-8 arguments, the result *must* be used intelligently**.**

- `MATCH` and some other functions are equally useful, if used properly. For instance, accompanying Clarion code uses `MATCH` in a regular expression that validates UTF-8 strings.
- `UPPER` and `LOWER` operate only on ASCII codes. They are generally safe for use with UTF-8, because they do not change any codes but ASCII. Typically, these are used to support case-insensitive processing of ASCII. Those who need the ability to change the case of non-ASCII alphabets need specialized functions to perform that task.
- Some Clarion string functions are best used in conjunction with helper functions. For example, `SUB(B,`*`start`*`,`*`length`*`)` may or may not return a valid UTF-8 substring from a valid UTF-8 string depending on the start and length values. To avoid this kind of complication, UTF-8 aware functions should be used. Among the attachments is a sampler of functions suitable for use with UTF-8 in Clarion `STRING`s, including a UTF-8 version of `SUB`.

The major consideration in all the above is to minimize (or eliminate) transitional pain for all parties including Soft Velocity and the existing user base.

Collation

Collation is: (a) The assembly of written information into a standard order; (b) The logical ordering of character strings according to defined precedence rules.

In Clarion, collation order is determined by the `CLACOLSEQ` entry in an ENV file. It determines the precedence used in all string comparisons, the sorting of strings in `QUEUE`s, and the ordering of database keys and indexes. Clarion's collation scheme presupposes a universe of one-byte characters, whose semantics are determined by the `CHARSET`.

The question obviously arises, how does Unicode fit into this? And the answer of course is that it doesn't.

The Unicode Collation Algorithm defines a procedure by which a Unicode collation order is determined for a particular purpose, but does not, and cannot impose an ironclad universal ordering of Unicode code points. The same code points are arranged in different sequences by different languages and cultures. Some languages do not give some vowels any weight in sorting, while most others do. There is no obvious reason for Latin alphabetic characters to precede or follow Greek or Cyrillic or any other alphabet. East Asian languages are not based on alphabets.

The UCA defines a means whereby a subset of code points may be defined, arranged in a locally desired sequence, using a complicated system of weights. (This is ultimately required by the complex sorting rules of some languages) All unspecified Unicode code points receive a lesser weight, but their sequence is predefined by the UCA. This scheme more or less resembles the Clarion `CLACOLSEQ` means of defining a collating sequence, with the obvious difference that Clarion's collating sequence is limited to a "character set" consisting of 255 one-byte characters.

It was recommended earlier in this document that Unicode not be supported in database keys and indexes, thus avoiding the collation issue in those particular instances. The same rules must logically be applied to string comparison and `QUEUE` sorting. This document proposes the environment file scheme currently used by Clarion not be changed, meaning Unicode sorting will not be directly supported.

If an application finds it necessary to collate Unicode strings, the suggested technique is to programmatically derive a key from the data to be sorted, where the key is used only to sort each record, but only for that purpose. Meanwhile, the sorted data would remain in Unicode format. If the desired collating sequence is a currently active code page then API functions can be used to derive the sorting key. This requires two stages: `MultibyteToWideChar` can translate UTF-8 to UTF-16. `WidecharToMultibyte` can then be used to translate the UTF-16 format to the encoding specified by `SYSTEM{PROP:Charset}`. The latter function can be instructed to replace any code points outside the desired character set with a default substitution character.

If the desired collating sequence is more complicated, various techniques are available for creating sorting keys.

Internal Considerations

This section will address the extent of the internal changes required to support the suggested externals described in the previous section.

- An application setting must be implemented that declares the application is a Unicode application
- The IDE must somehow prevent the insertion of the "UTF-8 signature" into Clarion-specific file types
- All controls with a specified `CHARSET` should continue to behave as currently implemented, i.e., no change.
- In a Unicode application, text assigned to a control without specifying a character set must be interpreted as UTF-8 format. The only new wrinkle this imposes on the runtime is to translate from UTF-8 to UTF-16, rather than from OEM to UTF-16. The API function in either case is `MultibyteToWideChar`. The translation from UTF-8 merely requires specifying CP_UTF8 as the code page.

However, in the case of OEM source, the character count can be obtained directly from the length of the input string, whereas in the case of UTF-8 this does not apply. This is an important distinction, because the function must be passed an array of sufficient size to contain the output. As it happens, the function can be used in a "calculate" mode which does no actual translation, but instead returns the size of the required buffer. The correct programming technique is therefore to first call the function to obtain the size, then allocate the buffer and call the function again to do the translation.

- Similarly, `TEXT` and `ENTRY` controls with no specified `CHARSET` are required by this proposal to be stored as UTF-8. Therefore, instead of translating from UTF-16 to OEM, the translation must be from UTF-16 to UTF-8. The same two-pass procedure should be used with `WidecharToMultibyte` as is used with `MultibyteToWideChar`. Call first to determine the length needed for allocation, then call again to translate.
- Because of security concerns that arise when invalid text is entered, UTF-8 strings should be validity checked. Windows provides a checking function, but warns that it is a statistical function. In the accompanying code, I present a reliable Clarion function to determine whether a string contains nothing but valid UTF-8. This is easily portable to C.
- The following page is a reference to the functions needed to manage UTF-8 to UTF-16 translations and vice-versa:
    - http://msdn.microsoft.com/en-us/library/dd318672%28VS.85%29.aspx
- To my knowledge, no other Clarion changes are required to implement Unicode support.

Summary

I hope in this article I have contributed at least a little bit toward a better understanding by the Clarion community of what Unicode is, and what it isn't.

Unicode is the accepted standard for developing Windows programs, and has been for several years. Lacking the ability to display multiple linguistic scripts simultaneously on the same Window, in the manner illustrated by the table containing various scripts shown in the section "Why should Clarion support Unicode?", a Clarion programmer is greatly disadvantaged. Unicode also provides clear advantages to those programmers who only work in their own language. The ability to easily and naturally incorporate a vast array of non-alphabetic symbols into standard text areas of programs (window titles, column headers, etc.) will quickly be appreciated. This also includes various standard superscripts and subscripts, such as the ability to write standard chemical notation in normal text. This is only one of many examples.

I have explained that Unicode has several completely interchangeable encodings, and I have argued that UTF-8 in particular should be adopted as the standard Unicode representation in Clarion programs, as opposed to any of the other possible representations. Using UTF-8, Unicode can be easily handled as text in databases, without necessarily providing Unicode support in keys or indexes. It is easily manipulated in Clarion programs with existing built-in string functions, and easily handled by the Clarion 7 integrated text editor. UTF-8 is the standard format for XML, as well as for the Internet, both in html and email. The ability to handle such communication naturally is a major advantage for any Windows developer. None of the other Unicode representations has any advantage to a Clarion programmer.

I have also described how easily UTF-8 support can be integrated into Clarion, by the simple convention of declaring a program to the IDE as a Unicode program in one setting, and having input/output text that is not specifically assigned a `CHARSET` be treated as UTF-8 by the runtime. Internally, providing Unicode support is as simple as identifying UTF-8 strings to the WinAPI as being UTF-8 format, and other strings as being some `CHARSET` format. This support is part and parcel of standard WinAPI functions for input and output of text, namely `MultibyteToWdeChar` and `WideCharToMultibyte`.

In short, I have described what Unicode is, why Clarion (and the Clarion community of developers) needs it, and very importantly, how to easily get from where we are to where we need to be.

Download the source

Phil Carroll has been twiddling bits on computers for about as long as there have been computers. His company, Enabling Simplicity LLC, presently offers primarily third-party add-ons to the Clarion market, with the flagship product UltraTree first appearing in 1996. Phil has always been more fascinated by mathematics, codes, algorithms, systems, and technologies than by developing applications and application databases. While at IBM in an earlier life, he at various times wrote embedded software in assembler for a variety of Intel and IBM microprocessors, helped design and implement peripheral subsystems, and built stochastic queuing models in APL to analyze the performance of IBM mainframe storage architectures. He discovered Clarion and the magic of Clarion templates in 1987, while looking for a high level language to develop applications for his wife's business needs. Since the introduction of Clarion for Windows, Clarion and the Clarion template language have been his primary programming languages. He was an early convert to, and advocate of, Clarion ABC. Phil's outside interests include listening to his collection of jazz music, going to jazz and wine festivals with his wife Kathryn, following a variety of sports on TV, and keeping up with science, politics and world events via a few selected periodicals and the Internet.

Article comments

BACK TO TOP

CLARION MAGAZINE
READ. LEARN. SOLVE.

Home | Subscribe | E-Books | News | Blog | Store | My ClarionMag | My Lists | Contact

What Is Unicode?

By Phillip Carroll

Posted April 30 2010

Clarion support for languages and characters other than English has historically been based on the support provided in Windows 95. This approach to the variety of scripts used by the languages of the world actually began well before Windows 95. It is exemplified by "character sets" (emphasis on plural), each of which is mapped onto the identical small set of codes. As a result, a single code has a multitude of interpretations, where the meaning of a code depends on the currently active "character set". Until not too long ago, character sets were normally an integral part of output devices such as monitors and printers.

In this universe, English and Western European languages cannot co-exist with Hebrew, or Greek, or any other language that uses a different script, nor can any of those languages co-exist with each other. In other words, multilingualism is impossible. The scheme even encounters problems with a single language script. Several competing Japanese encodings exist for the same Japanese characters, and the same applies to Russian, Polish and several other languages.

Windows 95 character sets are close relatives of the ISO 8859 standard character sets, but differ slightly. Such an environment eventually and inevitably results in a phenomenon known as "mojibake".

As early as 1989, a move was underway to establish a "universal character set" to replace this bewildering array of encodings. This has evolved into a standard known as Unicode. All Microsoft operating systems offered since NT have used Unicode as the internal representation of all text. ISO 8859 was abandoned in 2004 by dissolution of the working group. Although Microsoft continues to support the Windows 95 character sets, it does so only for legacy support reasons.

Periodically, the question will appear in various Clarion contexts: When will Clarion support Unicode? The ensuing conversation inevitably reveals a general state of understanding of Unicode that is remindful of the fable of the blind men asked to describe an elephant: "It's a tree," says one; "It's a rope," says another...

One such recent conversation led me to undertake some in-depth research into the mysteries of this thing called Unicode.

In this two-part article I will attempt to explain what Unicode is, why Clarion needs to support it, how to implement Unicode support in an evolutionary manner, and discusses the ramifications of that implementation from the perspective of both implementers and Clarion users.

What is Unicode?

From the Unicode topic on Wikipedia:

> "Unicode is a computing industry standard for the consistent representation and manipulation of text expressed in most of the world's writing systems."

From MSDN:

> "Compared to older mechanisms for handling character and string data, Unicode simplifies software localization and improves multilingual text processing. By using Unicode to represent character and string data in your applications, you can enable universal data exchange capabilities for global marketing, using a single binary file for every possible character code. Unicode does the following:
>
> - Allows any combination of characters, drawn from any combination of scripts and languages, to co-exist in a single document.
> - Defines semantics for each character.
> - Standardizes script behavior.
> - Provides a standard algorithm for bidirectional text.
> - Defines cross-mappings to other standards.
> - Defines multiple encodings of its single character set: UTF-7, UTF-8, UTF-16, and UTF-32. Conversion of data among these encodings is lossless.
>
> Unicode supports numerous scripts used by languages around the world, and also a large number of technical symbols and special characters used in publishing. The supported scripts include, but are not limited to, Latin, Greek, Cyrillic, Hebrew, Arabic, Devanagari, Thai, Han, Hangul, Hiragana, and Katakana. Supported languages include, but are not limited to, German, French, English, Greek, Russian, Hebrew, Arabic, Hindi, Thai, Chinese, Korean, and Japanese. Unicode currently can represent the vast majority of characters in modern computer use around the world, and continues to be updated to make it even more complete.

Unicode history

As a result of the situation described in the introduction, in 1989, the International Organization for Standardization (ISO) set out to compose a universal character set, and published a draft of ISO 10646 in 1990, known as the Universal Character Set standard. This draft standard provided for 679,477,248 possible characters in a complicated scheme of groups, planes, rows, and cells which was more or less an attempt to combine the existing character sets.

In 1990, two initiatives for a universal character set existed: Unicode, which at that time allocated 16 bits for every character, resulting in 65,536 possible characters, and ISO 10646. Software companies rebelled at the complexity and size requirement of the ISO standard and were able to convince a number of ISO National Bodies to vote against it. The ISO realized it could not continue to support the standard in its then-current state and negotiated the unification of its standard with Unicode. Since 1991, the Unicode Consortium has worked with ISO to develop The Unicode Standard ("Unicode") and ISO/IEC 10646 in tandem.

The organizing principle of Unicode is to assign a linear code point to a "character" as an abstract entity, separate from its visual appearance, namely, a "glyph". Thus, characters may be shown in hundreds of different "fonts", which are collections of glyphs in a particular style of writing. Another organizing principle is to treat alphabetic characters as independent of language. This principle would seem to suggest that Latin, Greek, and Cyrillic should share code points. Most Latin vowels and consonants are found in Greek and Cyrillic and are pronounced similarly. But, for mainly historic and cultural reasons, each of these alphabets has been assigned totally distinct sets of code points. (It is also obvious that the shapes of glyphs are radically different, not just stylistically different.) Nevertheless, each of these sets are shared by multiple languages. These organizing principles fit well with western languages.

In contrast to the treatment of Latin, Greek, and Cyrillic as noted above, Unicode adopted Han Unification as an organizing principle for the Asian ideographic languages. This approach is commonly referred to as "Unihan". This attempts to cover all of the Asian ideographic languages with a single set of code points. This creates many technical issues for various Asian languages, too technically complicated to cover in detail here. Suffice to say, Unicode is very controversial in East Asia.

The current standard

Today, both Unicode and the UCS standard define a code space of 1,114,112 "code points", mapped to integers in the hexadecimal range 0 to 10FFFF. (By comparison, ASCII has only 128 code points, and Extended ASCII has 256 code points.) In discussion, code points are referred to by writing "U+" followed by a hexadecimal number. The range from 0 to FFFF is called the Basic Multilingual Plane (BMP). Code points in the BMP are referred to by using four digits, i.e.: U+044F (which corresponds to the Cyrillic lower case Ya: я). Code points beyond the BMP range are referred to using five or six digits as necessary. Such as: U+437A2 or U+108FA2.

Although the standard provides for more than a million code points, as of today only slightly more 100,000 have been assigned.

Code points define unique identifiers, not computer encodings. Unicode has defined several encodings, only three of which are of primary interest to Clarion users:

UTF-32 – a 32-bit, fixed-width encoding

Because the entire set of Unicode code points is a 21-bit space, the UTF-32 "encoding" assigns code units in a one-to-one correspondence with the non-negative integers with the same values. Thus code point U+0000 is encoded as the number 0, U+FFFF is encoded as 65535, and U+10FFFF is encoded as the number 1,114,111. UTF-32 encoding is primarily used on Unix-class systems. I have mentioned it primarily just to note its existence.

UTF-16 – a 16-bit, variable-width encoding

UTF-16 characters consist of either one or two 16-bit values. The BMP range is encoded with a single 16-bit value identical to the least significant two bytes of UTF-32. Code points above U+FFFF are encoded as "surrogate pairs" of 16-bit values in the range D800 to DFFF. This is made possible by the fact that Unicode and ISO/IEC 10646 do not, and will never assign characters to any of the code points in the U+D800–U+DFFF range.

UTF-16 is of interest primarily because it is the basis of Windows internal support of Unicode.

UTF-8 – an 8-bit, variable-width encoding

A principal feature of UTF-8 is that it was designed to maximize compatibility with ASCII. The code points U+0000 to U+007F contain the 7-bit ASCII character set, and UTF-8 encodes these characters identically to ASCII. Therefore, a string of UTF-8 characters that contains only ASCII codes is identical to the corresponding ASCII string.

Byte values in particular ranges have the following pattern:

- 00-7F – A single byte character (ASCII)
- 80-BF – Tail byte of a multibyte character
- C0-C1 – Not allowed
- C2-DF – First byte of a two-byte character encoding
- E0-EF – First byte of a three-byte character encoding
- F0-F4 – First byte of a four-byte character encoding
- F5-FF – Not allowed

The observant reader may have noticed that the number of leading consecutive one-bits of valid encodings indicate whether the byte begins a character and if so, the length of the character. A single leading one-bit indicates a tail byte of a multibyte character. A leading zero-bit indicates an ASCII character. In all other cases, the number of leading one-bits is the length of the character in bytes.

These properties are not accidental. The encodings were specifically designed such that the non-ASCII code points contain no ASCII codes, and such that the initial byte of every character is distinguishable from a continuation byte of a multibyte character. Among other things, this guarantees that standard search libraries work as expected, and that existing string functions can be used to process UTF-8.

Observations on encodings

A UTF-16 string must use a pair of bytes for each code, which introduces potential problems:

- The different byte order of big-endian and little-endian computers requires that a means of detecting the ordering must be added to the protocol, such as with a byte order mark
- If a single byte is missing from a UTF-16 string, the rest of the string will be meaningless text.

A missing byte in UTF-8 affects only one character, not the entire string. Most communication and storage was designed for a stream of bytes, not words. As result of these qualities, UTF-8 is rapidly being adopted as the global standard encoding for both the Internet and for text stored in files.

The most compact encoding format to use depends on the languages involved. Consider a large volume of data currently stored in ASCII format, which is to be converted to Unicode. Conversion to UTF-8 requires no additional space. Files (containing only ASCII) encoded in any other Unicode format require special programming and occupy anywhere from two to four times as much space.

Text in all languages using code points below U+0800 (which includes all modern European languages) will be smaller in UTF-8 than UTF-16 due to the presence of ASCII spaces, newlines, numbers, punctuation, and mostly Latin letters. This set of code points represents 2048 characters. East Asian languages require a much larger code space, resulting in longer encodings in UTF-8 and consequently more space than UTF-16.

Fonts and glyphs

It is important to note, as explained earlier, that the ISO/IEC 10646 standard deals with characters as semantic entities. It has nothing to do with how the characters are visually represented in glyphs. The visual appearance of a code point is determined by the selection of font. Thus, the letter A can look like A, A, A, A or many other ways depending on the font chosen to display it.

To support all of the currently assigned code points would require a font with more than 100,000 glyphs. There is no such font in existence. Unlike the Universal Character Set, there is no such thing as a "universal font". However, a few fonts can display most of the world's modern languages. Among these are Tahoma and Lucida Sans Unicode.

Terminology

Because Microsoft support of Unicode dates to a time when the original 16-bit version of Unicode was still the standard, in much of Microsoft literature you will variously find Unicode described as a 16-bit encoding, or the term "16-bit Unicode" used incorrectly to describe the BMP subset of UTF-16. Such incorrect terminology unfortunately tends to promote a great deal of confusion about what the Unicode standard is.

There also exists an encoding known as DBCS, for "double byte character set" which has traditionally been used in CJK computing. (Chinese, Japanese, Korean) Some people incorrectly use "double byte" to mean the 16-bit UTF-16 encoding, or even the two-byte subset of UTF-8. Using this incorrect terminology leads to even more misunderstanding.

In summary:

- The Unicode/ISO 10646 standard provides the means to discard all previous notions of character sets and supplant it with the notion of a single unified character set.
- The character set can be encoded in several interchangeable computer codes with 1:1 mapping of code points.
- Internal text handling in all Microsoft systems since NT has been in the UTF-16 encoding of Unicode. (It should be noted that Windows support of code points beyond the BMP is spotty and varies by system.)
- UTF-8 is rapidly becoming the standard for communication, interchange, and storage.

In Part 2 of this article I'll discuss why Clarion should support Unicode.

Phil Carroll has been twiddling bits on computers for about as long as there have been computers. His company, Enabling Simplicity LLC, presently offers primarily third-party add-ons to the Clarion market, with the flagship product UltraTree first appearing in 1996. Phil has always been more fascinated by mathematics, codes, algorithms, systems, and technologies than by developing applications and application databases. While at IBM in an earlier life, he at various times wrote embedded software in assembler for a variety of Intel and IBM microprocessors, helped design and implement peripheral subsystems, and built stochastic queuing models in APL to analyze the performance of IBM mainframe storage architectures. He discovered Clarion and the magic of Clarion templates in 1987, while looking for a high level language to develop applications for his wife's business needs. Since the introduction of Clarion for Windows, Clarion and the Clarion template language have been his primary programming languages. He was an early convert to, and advocate of, Clarion ABC. Phil's outside interests include listening to his collection of jazz music, going to jazz and wine festivals with his wife Kathryn, following a variety of sports on TV, and keeping up with science, politics and world events via a few selected periodicals and the Internet.

Article comments

BACK TO TOP