A Clarion Magazine Reader

Clarion IDE Tips, Techniques and Tools

CLARION MAGAZINE
READ. LEARN. SOLVE.

Revision History

Oct 13 2014	First release

Clarion IDE

Assignment Alignment Addin Appeals

I'd never been one of those developers who carefully lines up assignment statements so all the equals signs are in the exact same column. Why not? Because it's always been way too much work! So when I first heard about Rick Martin's Assignment Alignment addin my reaction was a metaphorical shrug of the shoulders. I'd never been bothered to align assignment statements before. Why start now?

After all, what's wrong with this:

```
ArchiveUserQ.UserID = ausers:userid
            ArchiveUserQ.UserName = acm_users.username
            ArchiveUserQ.LowerUserName = lower(acm_users.username)
            ArchiveUserQ.FirstName = acm_users.firstname
            ArchiveUserQ.LastName = acm_users.lastName
            ArchiveUserQ.DaysExpired = acm_users.daysexpired
            ArchiveUserQ.DaysLeft = acm_users.daysleft
            ArchiveUserQ.Password = acm_users.PASSWORD
            add(ArchiveUserQ,ArchiveUserQ.Lowerusername)
```

On the other hand, if I can select some text, press Ctrl-Shift-I, and I get this:

```
ArchiveUserQ.UserID        = ausers:userid
            ArchiveUserQ.UserName      = acm_users.username
            ArchiveUserQ.LowerUserName = lower(acm_users.username)
            ArchiveUserQ.FirstName     = acm_users.firstname
            ArchiveUserQ.LastName      = acm_users.lastName
            ArchiveUserQ.DaysExpired   = acm_users.daysexpired
            ArchiveUserQ.DaysLeft      = acm_users.daysleft
            ArchiveUserQ.Password      = acm_users.PASSWORD
```

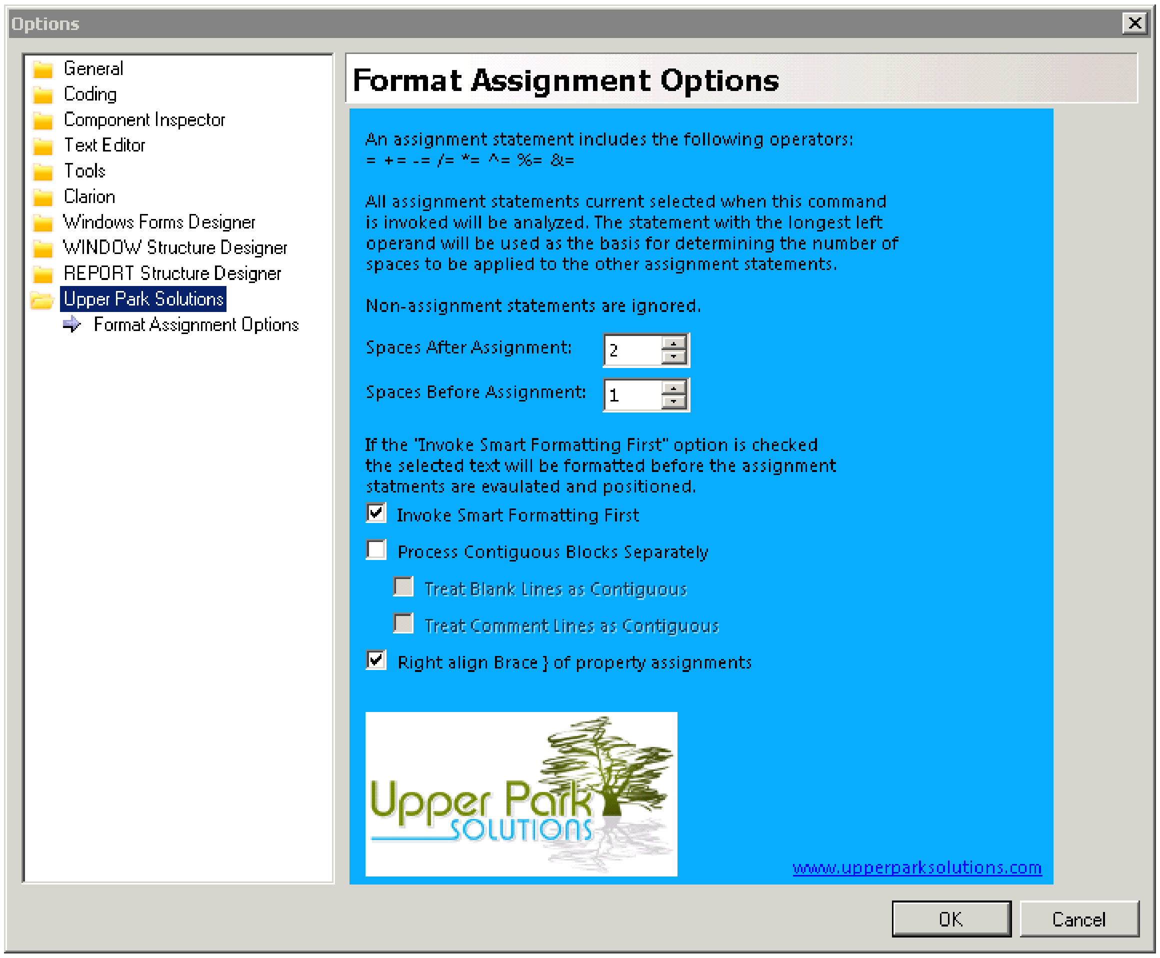
I have to admit that I find the second example cleaner and easier to read. For me it's not so much that I can see the variable names any better, it's simply that I can tell right away that all of these *are* assignment statements, and that's helpful. Also I work on a team with other developers, some of whom do format assignment statements this way, so if I can easily make my code conform to their standards life is easier.

But I noticed one problem early on. If I had a non-assignment statement somewhere in the middle of my code block, the indent wasn't preserved:

```
ArchiveUserQ.UserID        = ausers:userid
            ArchiveUserQ.UserName      = acm_users.username
            ArchiveUserQ.LowerUserName = lower(acm_users.username)
            ArchiveUserQ.FirstName     = acm_users.firstname
            ! A comment here
            ArchiveUserQ.LastName    = acm_users.lastName
            ArchiveUserQ.DaysExpired = acm_users.daysexpired
            ArchiveUserQ.DaysLeft    = acm_users.daysleft
            ArchiveUserQ.Password    = acm_users.PASSWORD
```

Notice that the alignment after the comment has been reset to the shortest required spacing.

I asked Rick about this, but I needn't have wasted his time. He just pointed me to the settings, which are available on the Options menu:



Format Assignment Options

An assignment statement includes the following operators:
= += -= /= *= ^= %= &=

All assignment statements current selected when this command is invoked will be analyzed. The statement with the longest left operand will be used as the basis for determining the number of spaces to be applied to the other assignment statements.

Non-assignment statements are ignored.

Spaces After Assignment: 2

Spaces Before Assignment: 1

If the "Invoke Smart Formatting First" option is checked the selected text will be formatted before the assignment statments are evaulated and positioned.

[✓] Invoke Smart Formatting First

[ ] Process Contiguous Blocks Separately

[ ] Treat Blank Lines as Contiguous

[ ] Treat Comment Lines as Contiguous

[✓] Right align Brace } of property assignments

www.upperparksolutions.com

I unchecked Process Contiguous Blocks Separately, and the addin maintained a consistent assignment operator column despite the insertion of a non-assignment statement. Slick!

```
ArchiveUserQ.UserID        = ausers:userid
            ArchiveUserQ.UserName      = acm_users.username
            ArchiveUserQ.LowerUserName = lower(acm_users.username)
            ArchiveUserQ.FirstName     = acm_users.firstname
            ! A comment here
            ArchiveUserQ.LastName      = acm_users.lastName
            ArchiveUserQ.DaysExpired   = acm_users.daysexpired
            ArchiveUserQ.DaysLeft      = acm_users.daysleft
            ArchiveUserQ.Password      = acm_users.PASSWORD
```

I left the Invoke Smart Formatting First option checked; I use Shift-I on a regular basis to reformat code, so I'm happy to have Rick's addin do that for me if I forget.

Now that I have this addin I use it constantly, both on new code and whenever I revisit old code.

The Format Assignment Statements Addin by Upper Park Solutions is available from ClarionShop. The price is a very reasonable $29.95. Highly recommended.

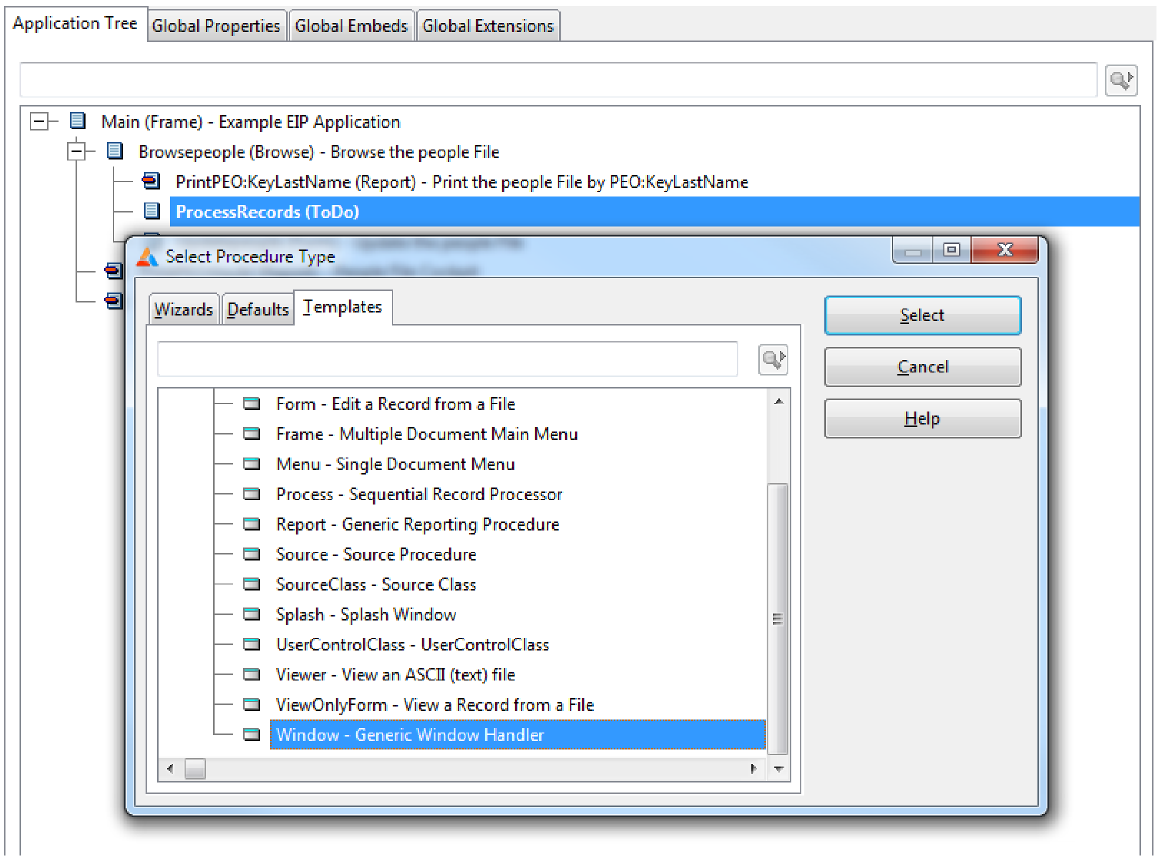
Creating a process without creating a process

Clarion's process template is a great tool for any kind of procedure that requires stepping through a filtered or unfiltered set of records. A little while ago I had a request to extract some data from a file, simply by stepping through records in primary key order, and my first thought was to go with a process template. But I also had a requirement to potentially restrict the records to a specific range of primary key values. So that meant popping up some kind of window to collect filter values.

As I was processing records in primary key order I decided I could do everything more simply with a window procedure. And it turned out to be even easier than I expected, thanks to some useful progress bar property statements. I'll get to those a little later.

In this article I'll show how to add a similar procedure to the venerable People.app.

I start by adding a button to the browse from which I want to call the export process. (I won't actually do any exporting in this example; instead I'll just display data from the records I process.) I set the Action of that button to call a new procedure, which I create using the Window template. Feel free to use the Window wizard if you're more comfortable with it - the only difference is that the wizard lets you set styles and will create the window for you; with the template you'll create the window the first time you go to the window editor.



Main (Frame) - Example EIP Application
Browsepeople (Browse) - Browse the people File
PrintPEO:KeyLastName (Report) - Print the people File by PEO:KeyLastName
ProcessRecords (ToDo)

Select Procedure Type

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

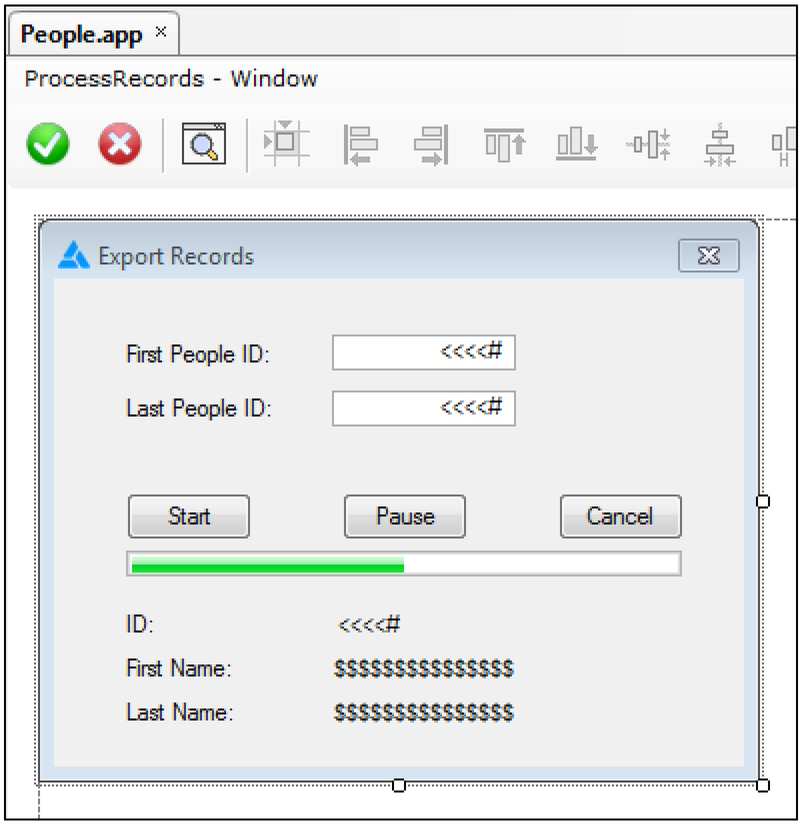
I need the following elements on the form:

- a starting primary key value
- an ending primary key value
- a progress bar
- a start button
- a cancel button

and just for fun:

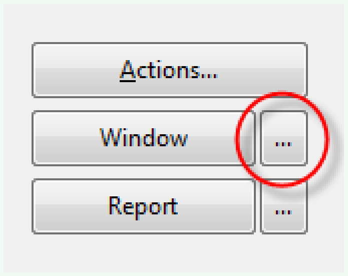
- a pause button

Here's the form with the UI elements complete:

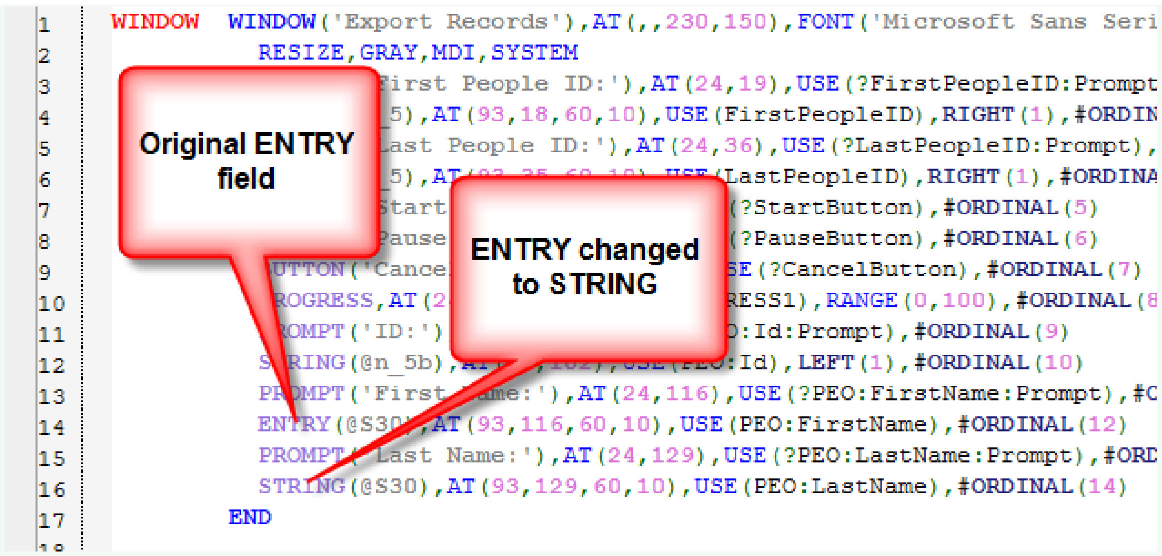


Since this process won't do anything other than loop through the records (the XML export that inspired the original procedure is outside the scope of this article) all I'm doing is displaying the current record on screen.

Changing control types
If you have entry fields you'd rather show as strings on a window, populate them as you normally would. Then use the ellipsis button to edit the window definition directly.



Simply change the ENTRY to a STRING - everything else on the line can stay the same.



```
WINDOW  WINDOW('Export Records'),AT(,,230,150),FONT('Microsoft Sans Seri
          RESIZE,GRAY,MDI,SYSTEM
                    irst People ID:'),AT(24,19),USE(?FirstPeopleID:Prompt
                    _5),AT(93,18,60,10),USE(FirstPeopleID),RIGHT(1),#ORDIN
                    ast People ID:'),AT(24,36),USE(?LastPeopleID:Prompt),
                    _5),AT                     (LastPeopleID),RIGHT(1),#ORDINA
                    Start                      (?StartButton),#ORDINAL(5)
                    Pause                      (?PauseButton),#ORDINAL(6)
                UTTON('Cance                  SE(?CancelButton),#ORDINAL(7)
                ROGRESS,AT(2                  RESS1),RANGE(0,100),#ORDINAL(8
                ROMPT('ID:')                  :Id:Prompt),#ORDINAL(9)
          S RING(@n_5b),                        :Id),LEFT(1),#ORDINAL(10)
          PR MPT('First        me:'),AT(24,116),USE(?PEO:FirstName:Prompt),#C
          ENTRY(@S30  ,AT(93,116,60,10),USE(PEO:FirstName),#ORDINAL(12)
          PROMPT(  Last Name:'),AT(24,129),USE(?PEO:LastName:Prompt),#ORD
          STRING(@S30),AT(93,129,60,10),USE(PEO:LastName),#ORDINAL(14)
        END
```

Once you get comfortable with Clarion Window structures you can make all kinds of changes. One I make regularly is change entry fields to spin boxes. It's less work than populating a new control and changing the use variable.

Using a timer

Looping through records in a tight loop with no possibility of interrupting the loop is almost never a good idea unless you are only dealing with a handful of records. Clarion reports and processes avoid this problem by using a window timer and looping through a smaller number of records on each timer event. In between these processing loops the UI can respond to user input.

I'm going to use a similar approach, which means I need to be able to put code into the Event:Timer embed. And that embed only appears when the window has a Timer attribute. Any non-zero value will do.

Setting up for sequential processing

When the window opens up, any timer event code will execute because of that required Timer attribute. So I need to disable the timer. Right after Self.Open(WINDOW) I add this code:

```
0{prop:timer} = 0
```

The value of 0 before the property assignment means I am assigning to the currently active window. I could also use

```
Window{prop:timer} = 0
```

but "0" takes less typing. And if you're ever writing code that needs to apply generically to a window using 0 means you don't need to know the window label.

I have a little more setup code that executes around the same time (actually just before opening the window):

WindowManager Method Code Section priority 7800

```
FirstPeopleID = PEO:ID
    clear(PEO:ID,1)
    set(PEO:KeyId,PEO:KeyId)
    previous(people)
    if errorcode()
        LastPeopleID = 0
    else
        LastPeopleID = PEO:Id
    end
    clear(PEO:RECORD)
```

This code:

- assumes the first record I want to process is in memory, and saves the ID of that record in FirstPeopleID
- primes LastPeopleID with the hightest ID in the file
- clears the record so nothing displays when the window first opens

Using timer events

Here's the Event:Timer code (I'll explain the use of PROP:Progress in a moment):

```
next(People)
        if errorcode() or PEO:Id > LastPeopleID
            0{prop:timer} = 0
            post(event:closewindow)
        else
            ?Progress{PROP:Progress} = PEO:Id
        end
```

When the start button is clicked this code executes:

?StartButton, Control Event Handling priority 8500

```
TimerInterval = 50
        PEO:Id = FirstPeopleID
        set(PEO:KeyId,PEO:KeyId)
        0{prop:timer} = TimerInterval
        ?Progress{PROP:RangeLow} = FirstPeopleID
        ?Progress{PROP:RangeHigh} = LastPeopleID
        disable(?StartButton)
        enable(?PauseButton)
```

First I set a local variable which is the value that will be applied to Prop:Timer. There are two places this happens (as you'll see); I use a local variable so that if I want to change the value I only need to do so in one place. A value of 50 means the timer will execute every half second, which is probably a lot less frequent than I'd really want it to happen but it allows the process to execute at a good demonstration speed. I'll also only be processing one record per timer event.

I then do a Set in primary key order on the People file.

Displaying the progress bar

This is the bit I really like about this whole procedure. By default, progress bars have a range of values from 1 to 100, and if you want to use that range and you don't have exactly 100 records you need to do some math.

```
?Progress{PROP:RangeLow} = FirstPeopleID
?Progress{PROP:RangeHigh} = LastPeopleID
```

The RangeLow and RangeHigh properties let you set the lower and upper boundaries to whatever value you like. Because I'm setting the low value to the first primary key value, and the high value to the last primary key value, all I need to do to display the progress bar is set PROP:Progress to the current primary key value (as shown earlier):

```
?Progress{PROP:Progress} = PEO:Id
```

I'm not sure just how big RangeHigh can get, but I've tried it with the maximum LONG values and it still works, so if you're using LONGs for primary key values you needn't worry.

Pause and Cancel

There are two more events I need to respond to: Pause and Cancel.

The Cancel button code is simple - just disable the timer and post a CloseWindow event:

```
0{prop:timer} = 0
        post(EVENT:CloseWindow)
```

The Pause button code is more elaborate, as I want to be able to toggle the button text between 'Pause' and 'Resume'. This is also the second place where I use the TimerInterval variable.

```
if ?PauseButton{prop:text} = 'Pause'
            if 0{prop:timer} > 0
                0{prop:timer} = 0
                ?PauseButton{prop:text} = 'Resume'
            end
        else
            ?PauseButton{prop:text} = 'Pause'
            0{prop:timer} = TimerInterval
        end
```

Summary

Although I could have used a Process template in this situation, with relatively little code I have achieved the following:

- Easy selection of range limiting values
- Accurate progress bar display
- The ability to pause the process

This technique works best on sequentially numbered single component keys, such as primary keys, although with a bit of work you might be able to adapt it to other situations.

And of course this was a one-off job; if I come across other places where I need something similar I'll look into creating a class and/or a template. But sometimes a little hand code goes a long way.

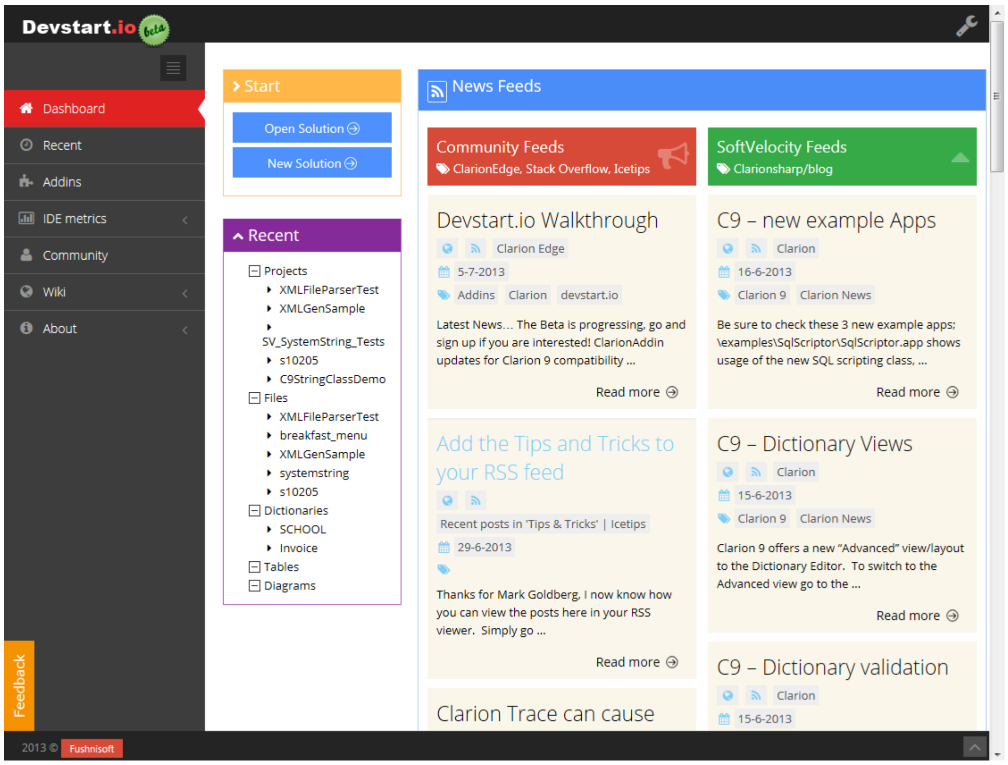
Source code

Download the C9.1 app (includes a TXA and DCTX)

Devstart.io - start page addin for Clarion 9

Brahn Partridge, who has created a number of Clarion IDE addins, has a new project in the works: Devstart.io. This addin is a replacement for Clarion 9's Start Page and employs a left-side menu to display multiple pages.

Here's the main Dashboard page:


Devstart.io beta

Dashboard
Recent
Addins
IDE metrics
Community
Wiki
About

Start
Open Solution
New Solution

Recent
- Projects
  - XMLFileParserTest
  - XMLGenSample
  - SV_SystemString_Tests
  - s10205
  - C9StringClassDemo
- Files
  - XMLFileParserTest
  - breakfast_menu
  - XMLGenSample
  - systemstring
  - s10205
- Dictionaries
  - SCHOOL
  - Invoice
- Tables
- Diagrams

News Feeds

Community Feeds
ClarionEdge, Stack Overflow, Icetips

SoftVelocity Feeds
Clarionsharp/blog

Devstart.io Walkthrough
Clarion Edge
5-7-2013
Addins Clarion devstart.io
Latest News… The Beta is progressing, go and sign up if you are interested! ClarionAddin updates for Clarion 9 compatibility …
Read more

C9 – new example Apps
Clarion
16-6-2013
Clarion 9 Clarion News
Be sure to check these 3 new example apps; \examples\SqlScriptor\SqlScriptor.app shows usage of the new SQL scripting class, …
Read more

Add the Tips and Tricks to your RSS feed
Recent posts in 'Tips & Tricks' | Icetips
29-6-2013
Thanks for Mark Goldberg, I now know how you can view the posts here in your RSS viewer. Simply go …
Read more

C9 – Dictionary Views
Clarion
15-6-2013
Clarion 9 Clarion News
Clarion 9 offers a new "Advanced" view/layout to the Dictionary Editor. To switch to the Advanced view go to the …
Read more

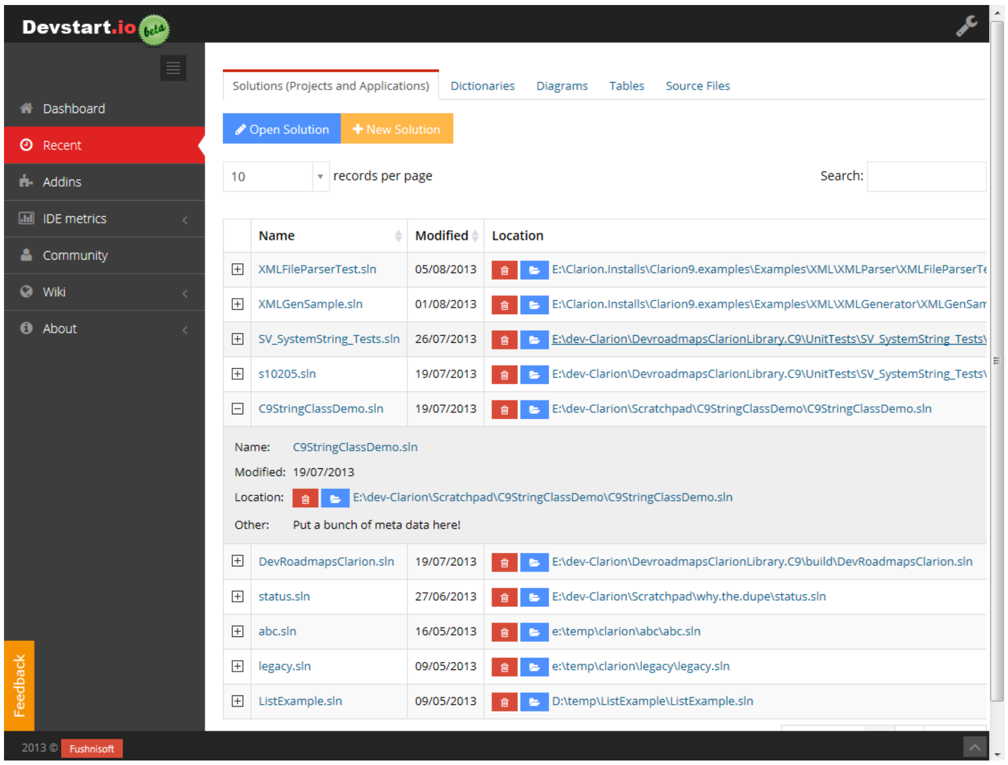
C9 – Dictionary validation
Clarion
15-6-2013

Clarion Trace can cause

Feedback

2013 © Fushnisoft

The Recents page features collapsible project information, a records per page drop list and a locator field:


Devstart.io beta

Dashboard
Recent
Addins
IDE metrics
Community
Wiki
About

Solutions (Projects and Applications) | Dictionaries | Diagrams | Tables | Source Files

Open Solution
New Solution

10 records per page
Search:

	Name	Modified	Location
⊞	XMLFileParserTest.sln	05/08/2013	E:\Clarion.Installs\Clarion9.examples\Examples\XML\XMLParser\XMLFileParserTe
⊞	XMLGenSample.sln	01/08/2013	E:\Clarion.Installs\Clarion9.examples\Examples\XML\XMLGenerator\XMLGenSam
⊞	SV_SystemString_Tests.sln	26/07/2013	E:\dev-Clarion\DevroadmapsClarionLibrary.C9\UnitTests\SV_SystemString_Tests\
⊞	s10205.sln	19/07/2013	E:\dev-Clarion\DevroadmapsClarionLibrary.C9\UnitTests\SV_SystemString_Tests\
⊟	C9StringClassDemo.sln	19/07/2013	E:\dev-Clarion\Scratchpad\C9StringClassDemo\C9StringClassDemo.sln

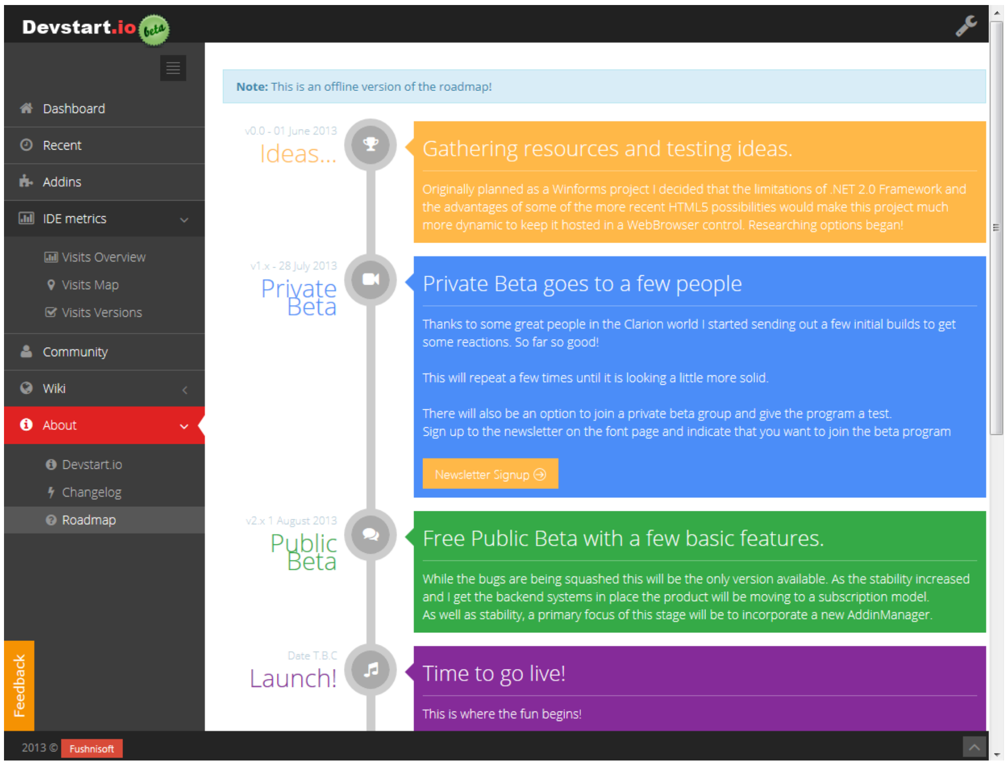
Name: C9StringClassDemo.sln
Modified: 19/07/2013
Location: E:\dev-Clarion\Scratchpad\C9StringClassDemo\C9StringClassDemo.sln
Other: Put a bunch of meta data here!

	Name	Modified	Location
⊞	DevRoadmapsClarion.sln	19/07/2013	E:\dev-Clarion\DevroadmapsClarionLibrary.C9\build\DevRoadmapsClarion.sln
⊞	status.sln	27/06/2013	E:\dev-Clarion\Scratchpad\why.the.dupe\status.sln
⊞	abc.sln	16/05/2013	e:\temp\clarion\abc\abc.sln
⊞	legacy.sln	09/05/2013	e:\temp\clarion\legacy\legacy.sln
⊞	ListExample.sln	09/05/2013	D:\temp\ListExample\ListExample.sln

Feedback

2013 © Fushnisoft

There's also information about Devstart.io itself, including the product roadmap:



Note: This is an offline version of the roadmap!

v0.0 - 01 June 2013
Ideas...

Gathering resources and testing ideas.

Originally planned as a Winforms project I decided that the limitations of .NET 2.0 Framework and the advantages of some of the more recent HTML5 possibilities would make this project much more dynamic to keep it hosted in a WebBrowser control. Researching options began!

v1.x - 28 July 2013
Private Beta

Private Beta goes to a few people

Thanks to some great people in the Clarion world I started sending out a few initial builds to get some reactions. So far so good!

This will repeat a few times until it is looking a little more solid.

There will also be an option to join a private beta group and give the program a test.
Sign up to the newsletter on the font page and indicate that you want to join the beta program

v2.x 1 August 2013
Public Beta

Free Public Beta with a few basic features.

While the bugs are being squashed this will be the only version available. As the stability increased and I get the backend systems in place the product will be moving to a subscription model.
As well as stability, a primary focus of this stage will be to incorporate a new AddinManager.

Date T.B.C
Launch!

Time to go live!

This is where the fun begins!

Many of the menu items are currently under construction, but what is there seems to work very well. Brahn continues to do great work extending the IDE for a better developer experience.

For more information visit http://devstart.io/

For more on Brahn's other cool IDE products go to http://clarionaddins.com/

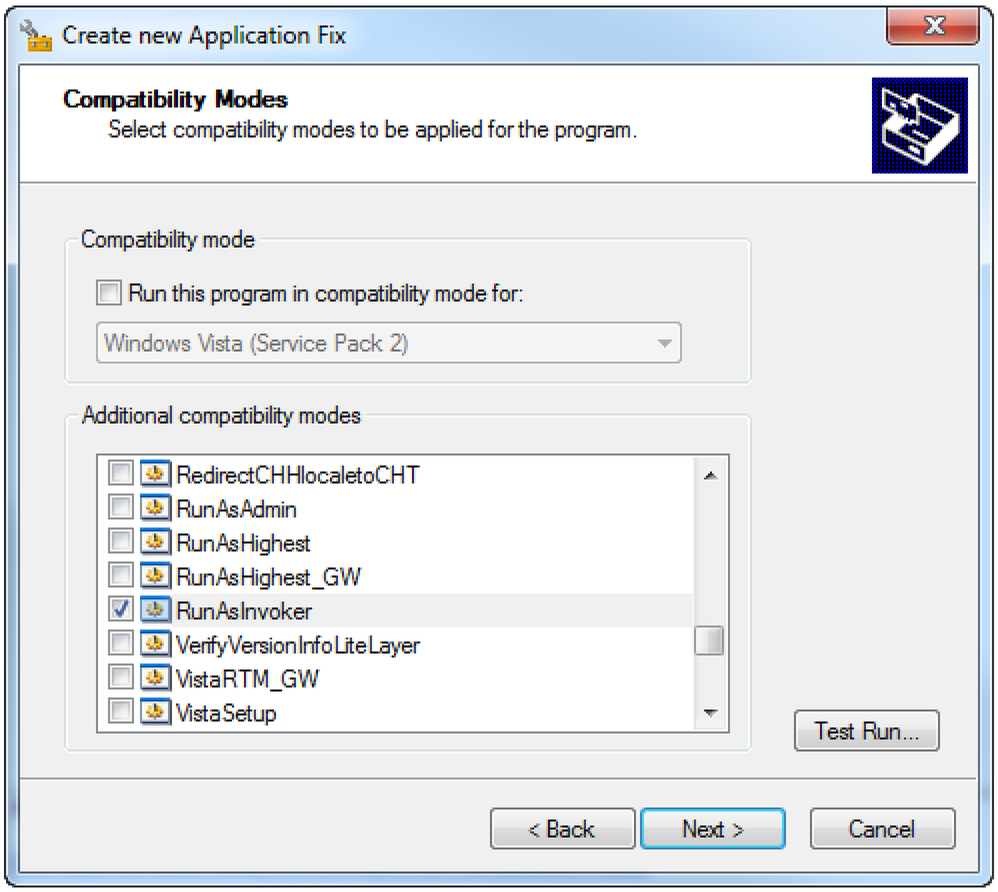
Disabling UAC for Clarion only

The last time I went through a full Windows install I gave in and left UAC enabled. It's no great hardship, and it does give me a greater feeling of security.

That is, it's no great hardship *except* for having to deal with the annoying UAC prompt every single time I run Clarion. Finally I got annoyed enough to start looking for a way to disable UAC for just the Clarion IDE.

I found the answer at http://www.wintuts.com/Disable-UAC-for-certain-applications. That page has a nice, detailed explanation of the process. Here's a summary.

First, download and install the Microsoft Application Compatibility Toolkit, then run the Compatibility Administrator. Under the Custom Databases node select the default custom database and right-click. Choose Create New|Application Fix. Fill in the blanks on that form and click Next. On the Compatibility Modes window select RunAsInvoker. Click Test Run if you want to be sure this works.


Create new Application Fix
Compatibility Modes
Select compatibility modes to be applied for the program.
Compatibility mode
Test Run...
< Back
Next >
Cancel

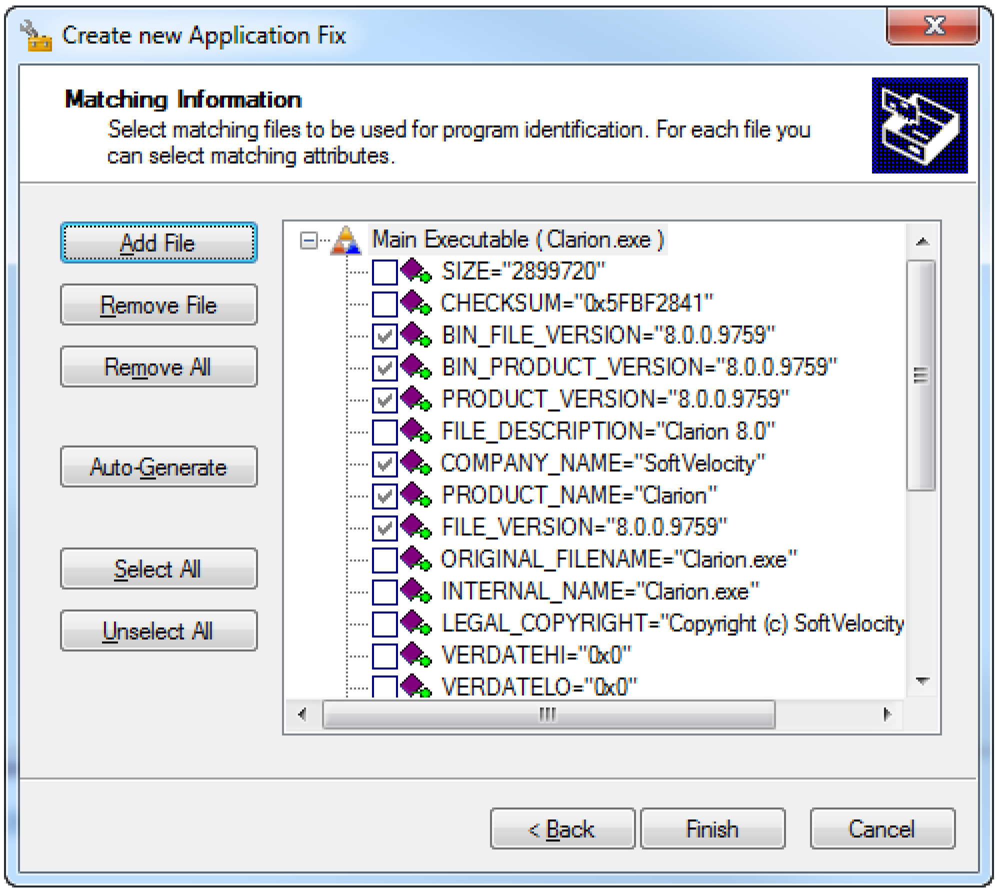
Run this program in compatibility mode for: [ ]
Windows Vista (Service Pack 2)

Additional compatibility modes

- [ ] RedirectCHHlocaletoCHT
- [ ] RunAsAdmin
- [ ] RunAsHighest
- [ ] RunAsHighest_GW
- [✓] RunAsInvoker
- [ ] VerifyVersionInfoLiteLayer
- [ ] VistaRTM_GW
- [ ] VistaSetup

You can accept the defaults on the following Compatibility Fixes window.

You may however want to consider some of the options for the Matching Information window. If you accept the defaults, then any time you upgrade Clarion you'll also have to recreate or update the database information. If you remove all of the matching options then you can upgrade without issue, but UAC will not be able to detect any malicious changes to the executable.



Matching Information
Select matching files to be used for program identification. For each file you can select matching attributes.

Main Executable (Clarion.exe)
- [ ] SIZE="2899720"
- [ ] CHECKSUM="0x5FBF2841"
- [x] BIN_FILE_VERSION="8.0.0.9759"
- [x] BIN_PRODUCT_VERSION="8.0.0.9759"
- [x] PRODUCT_VERSION="8.0.0.9759"
- [ ] FILE_DESCRIPTION="Clarion 8.0"
- [x] COMPANY_NAME="SoftVelocity"
- [x] PRODUCT_NAME="Clarion"
- [x] FILE_VERSION="8.0.0.9759"
- [ ] ORIGINAL_FILENAME="Clarion.exe"
- [ ] INTERNAL_NAME="Clarion.exe"
- [ ] LEGAL_COPYRIGHT="Copyright (c) SoftVelocity
- [ ] VERDATEHI="0x0"
- [ ] VERDATELO="0x0"

Click Finish, then use File|Save and save the database to disk. Run a command prompt elevated and execute

sdbinst *name_of_the_database_file*

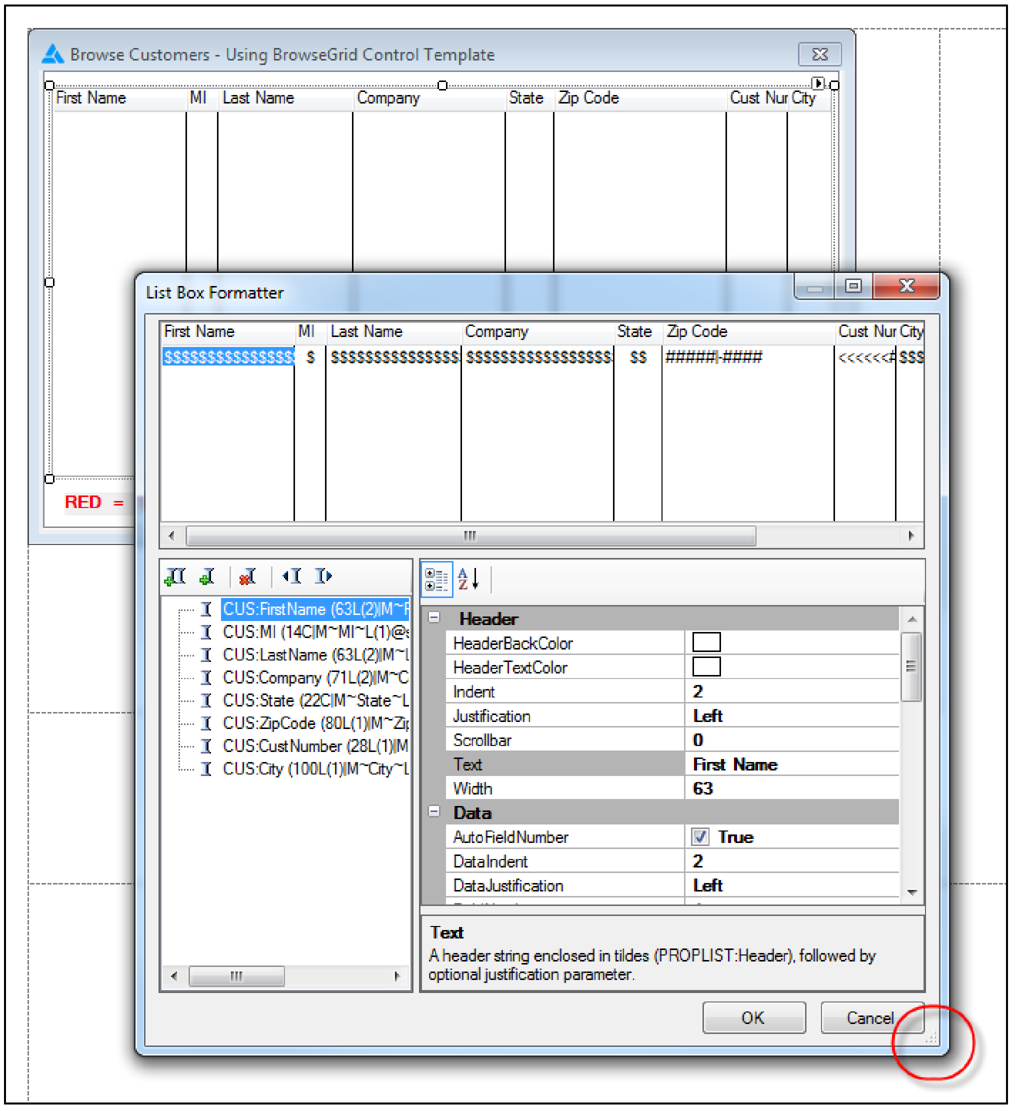
And that's it - if you did everything right you'll be able to run Clarion without getting a UAC prompt.

I tried to add two entries to my database, one for Clarion 8 and one for Clarion 9 but I got a warning that as the entries were similar I could get unpredictable results. I'm guessing that since the settings are the same that wouldn't be a problem, but to be safe I created two databases, one for each Clarion version.

Expanding and moving the listbox formatter and data property windows

Mike Hanson reminded me of a nifty Windows trick the other day, one which I'd never thought to use on the Clarion IDE's list box formatter.

The list box formatter window is by default quite small, but you probably already know that it's resizable, Just grab the handle at the lower right corner and drag the window to the desired size.


Browse Customers - Using BrowseGrid Control Template
First Name MI Last Name Company State Zip Code Cust Nur City
RED =
List Box Formatter
First Name MI Last Name Company State Zip Code Cust Nur City
OK
Cancel

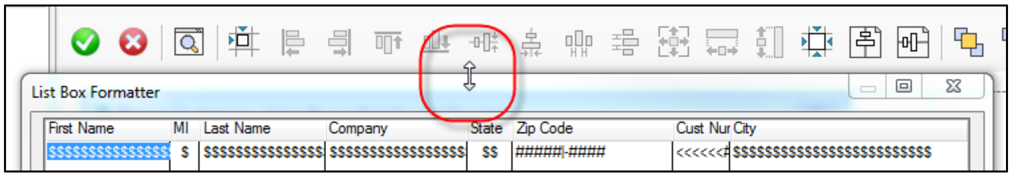
Header	
HeaderBackColor	⬜
HeaderTextColor	⬜
Indent	2
Justification	Left
Scrollbar	0
Text	First Name
Width	63
Data	
AutoFieldNumber	✓ True
DataIndent	2
DataJustification	Left

CUS:FirstName (63L(2)|M~F
CUS:MI (14C|M~MI~L(1)@s
CUS:LastName (63L(2)|M~L
CUS:Company (71L(2)|M~C
CUS:State (22C|M~State~L
CUS:ZipCode (80L(1)|M~Zi
CUS:CustNumber (28L(1)|M
CUS:City (100L(1)|M~City~L

Text
A header string enclosed in tildes (PROPLIST:Header), followed by optional justification parameter.

You can also maximize the window if you like, and the IDE does remember the last position of the window.

As of Windows 7, you can also maximize windows vertically. Just put your cursor at the top or bottom of the window until you see the resize arrow and then double-click (as Mike reminded me). Presto - the window maximizes vertically! This is especially useful because at the default size many of the properties are not visible - vertically maximizing the window shows as many properties as possible and greatly simplifies setting those properties.


List Box Formatter
First Name MI Last Name Company State Zip Code Cust Nur City

You can also do this via the keyboard. Hold down Shift+Windows and the up or down arrow key to vertically maximize/restore the window.

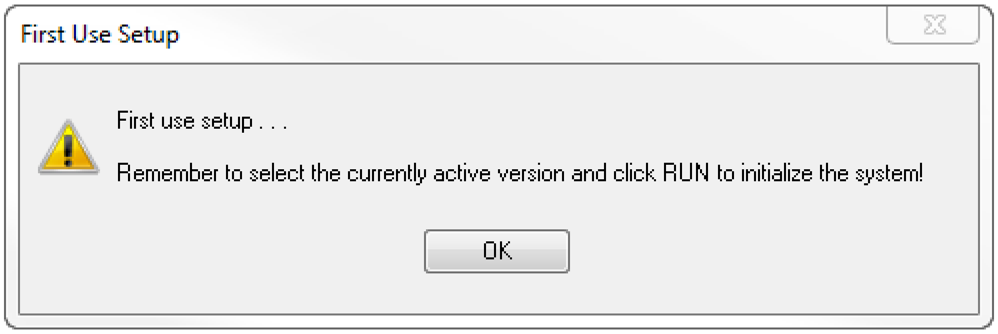
If you have multiple monitors, try Shift+Windows and the left and right arrow keys - these will move the list box formatter window to the next monitor. However you locate/resize the window the IDE will remember the location the next time you open the window.

Another window that behaves this way in the IDE (aside from the IDE itself) is the data properties window (as when you're in the data pad and you want to edit local, module or global data. Because the data properties window has a control properties pad similar to the one in the list box formatter, vertically maximizing is helpful her too although mainly just on the Controls tab.

Most IDE windows, however, do not follow these rules of behavior. Dockable windows such as the properties pad, task list etc. are unaffected by the double-click and Windows key shortcuts. These windows can still be relocated and resized using the mouse.

But maybe there are other windows in the IDE that can be do these Windows 7 tricks. If you know of any please post a comment!

Managing multiple Clarion versions with ClaSwitch2


First Use Setup

First use setup . . .

Remember to select the currently active version and click RUN to initialize the system!

Quick starts for classes

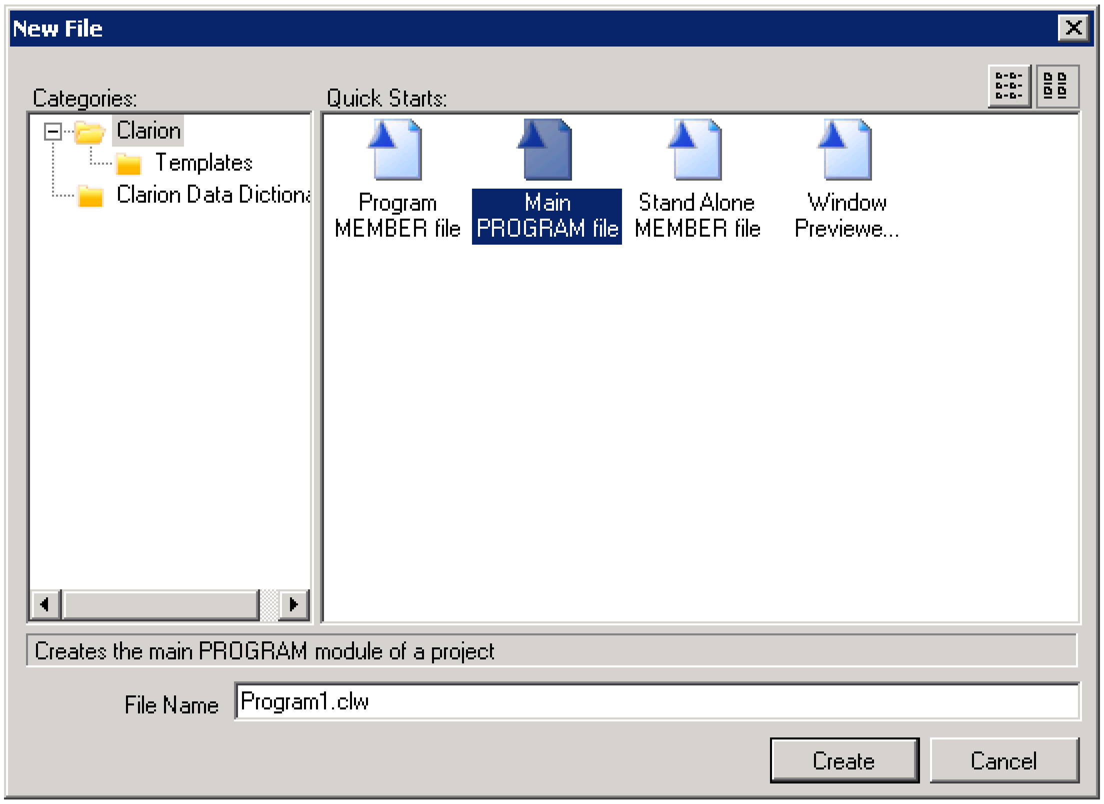
While working on The Problem with Embeds, Revisited I had to step through the process of using the Clarion IDE to create .INC and .CLW files to hold, respectively, class headers and class methods. I don't do this much any more; normally I use John Hickey's excellent ClarionLive Class Creator, which is part of the ClarionLive Utility Pack.

It's easy enough to create the .CLW file using just the IDE - simply use the Stand Alone Member file quick start as that produces a file with an empty Member() as well as a Map statement.

You can take the same approach when creating the .INC file, although you have to be careful to change the file extension. And of course you'll need to gut the quick-started file because it doesn't have anything you need. If you haven't written a lot of classes you may find it difficult to remember just what you *do* need.

While I think anyone who writes classes can benefit from the ClarionLive Class Creator, I also think that Clarion could use a Quick Start for classes. So I started digging.

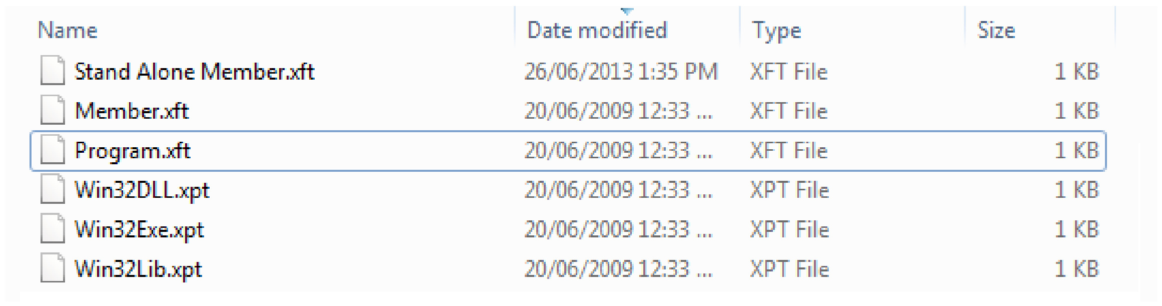
Here are the shipping Quick Starts:



Creates the main PROGRAM module of a project

File Name Program1.clw

I don't know where the Window Previewer comes from, but the other items can all be found in `bin\Addins\BackendBindings\ClarionBinding\ClarionWin\Templates` under the Clarion root directory. Files that end in .xft are text templates, files that end in xpt are project templates:



Name	Date modified	Type	Size
Stand Alone Member.xft	26/06/2013 1:35 PM	XFT File	1 KB
Member.xft	20/06/2009 12:33 ...	XFT File	1 KB
Program.xft	20/06/2009 12:33 ...	XFT File	1 KB
Win32DLL.xpt	20/06/2009 12:33 ...	XPT File	1 KB
Win32Exe.xpt	20/06/2009 12:33 ...	XPT File	1 KB
Win32Lib.xpt	20/06/2009 12:33 ...	XPT File	1 KB

Here's what Member.xft looks like:

```
<?xml version="1.0"?>
<Template author="SoftVelocity" version="1.0">
 <Config
    name        = "${res:Templates.File.Member.Name}"
    icon        = "CW.File.EmptyFile"
    category    = "Clarion"
    defaultname = "Member${Number}.clw"
    language    = "Clarion"/>

 <Description>${res:Templates.File.Member.Description}</Description>

 <Files>
  <File name="${FullName}" language="Clarion">
   <![CDATA[
  MEMBER('${ProjectName}')
${StandardHeader.Clarion}
  MAP
  END

]]>
  </File>
 </Files>
 <AdditionalOptions/>
</Template>
```

As you can see it's an XML file that contains some configuration information as well as a CDATA section containing the text to be placed in the file itself.

I created two XFT files, one for the class header and one for the class source. The CDATA section was easy to handle. But how to change the name of the quick start and it's description? I poked around a bit trying to find out where `res:Templates.File.Member.Name` was stored, but assuming I did find it that would probably mean at least one existing file that would need to be updated and/or replaced.

It was easier to replace these resource identifiers with literal strings. I also changed the defaultname attribute to a literal string.

Here's the source for ClassHeader.xft:

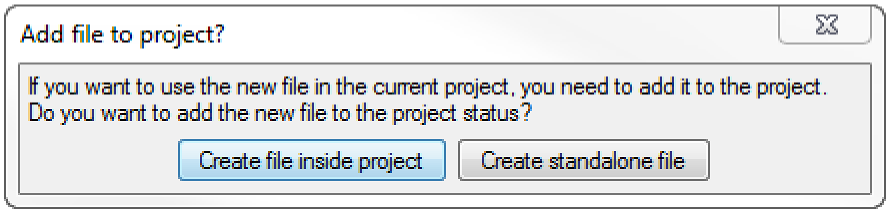
```
<?xml version="1.0"?>
<Template author="SoftVelocity" version="1.0">
 <Config
    name        = "Class Header"
    icon        = "CW.File.EmptyFile"
    category    = "Clarion"
    defaultname = "ClassName?.inc"
    language    = "Clarion"/>

 <Description>.INC file containing the class definition</Description>

 <Files>
  <File name="${FullName}" language="Clarion">
   <![CDATA[
${StandardHeader.Clarion}
!------------------------------------------------------------------------------------
! Change ClassName? to the name of your class
!------------------------------------------------------------------------------------
!------------------------------------------------------------------------------------
! Use of the ,Type attribute is recommended, although it means you'll
! have to instantiate the class before you can use it
!------------------------------------------------------------------------------------

ClassName?     class,module('ClassName?.clw'),link('ClassName?',1),Thread!,type
!ExampleMethod              procedure
      end

]]>
  </File>
 </Files>
 <AdditionalOptions/>
</Template>
```

And here's ClassMethods.xft:

```
<?xml version="1.0"?>
<Template author="SoftVelocity" version="1.0">
 <Config
    name        = "Class Methods"
    icon        = "CW.File.EmptyFile"
    category    = "Clarion"
    defaultname = "ClassName?.clw"
    language    = "Clarion"/>

 <Description>.CLW file containing the class's methods</Description>

 <Files>
  <File name="${FullName}" language="Clarion">
   <![CDATA[

${StandardHeader.Clarion}
!--------------------------------------------------------------------------------
! Change ClassName? to the name of your class
!--------------------------------------------------------------------------------

                                            Member()
                                            Map
                                            End

    include('ClassName?.inc'),once

!--------------------------------------------------------------------------------
! Implement your class methods here
!--------------------------------------------------------------------------------
!ClassName?.ExampleMethod        procedure
!  code
!  ! do whatever
]]>
  </File>
 </Files>
 <AdditionalOptions/>
</Template>
```

Inside project or standalone?

If you create a new file and you have a project open, you'll see this window:


Add file to project?

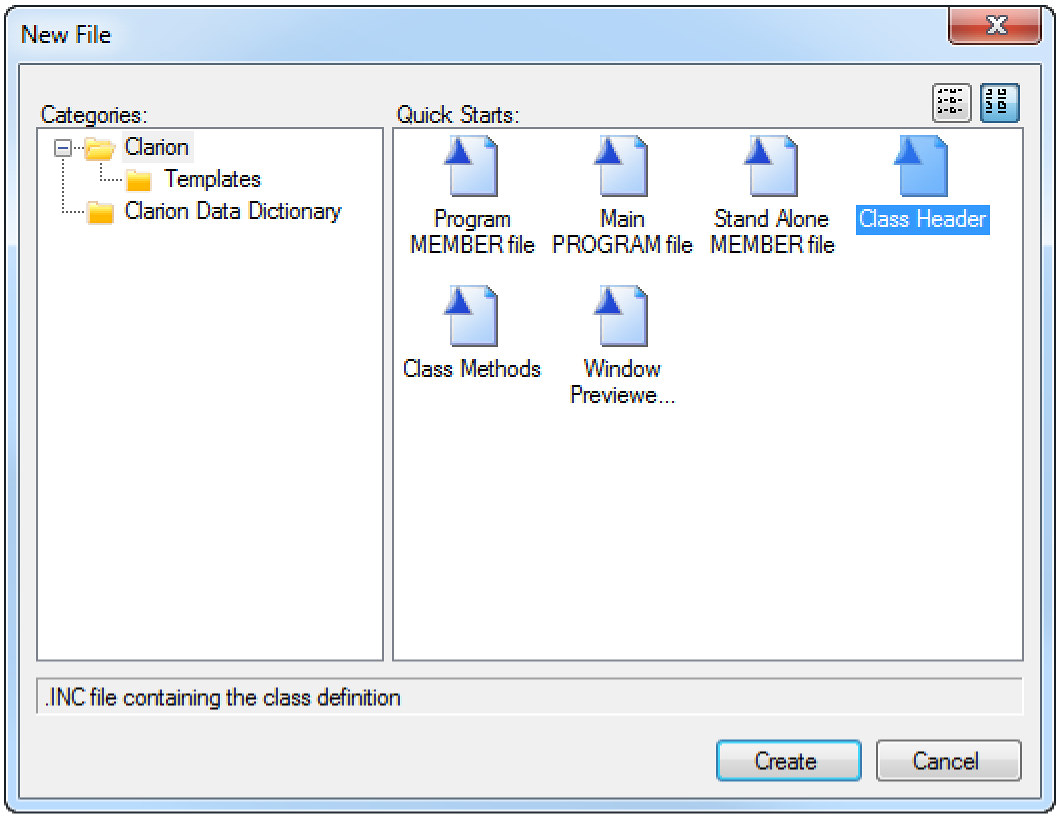
If you want to use the new file in the current project, you need to add it to the project.
Do you want to add the new file to the project status?

If you create the file inside the project it will be added to the project's list of files to compile. This is not necessary because the class declarations has a LINK attribute which will cause the class file to be compiled and linked.

In the case of INC files you don't want them in the project anyway. By default the compiler will complain because it doesn't know how to handle INC files, so you'll have to change its properties to Compile: Never. That's a hassle.

I recommend you create all class files as standalone files. Then (after you've saved the files) add them to your solution by right-clicking on the Solution in the Solution explorer and choosing Add | Add Item. To keep the files better organized you may want to create a new Solution folder first and then add the classes to that folder.

Here's the Class Header template in action, assuming you're creating the files as standalone. If you create them inside the project you'll also be able to override the file name here (although you'll still have to explicitly save the file later).

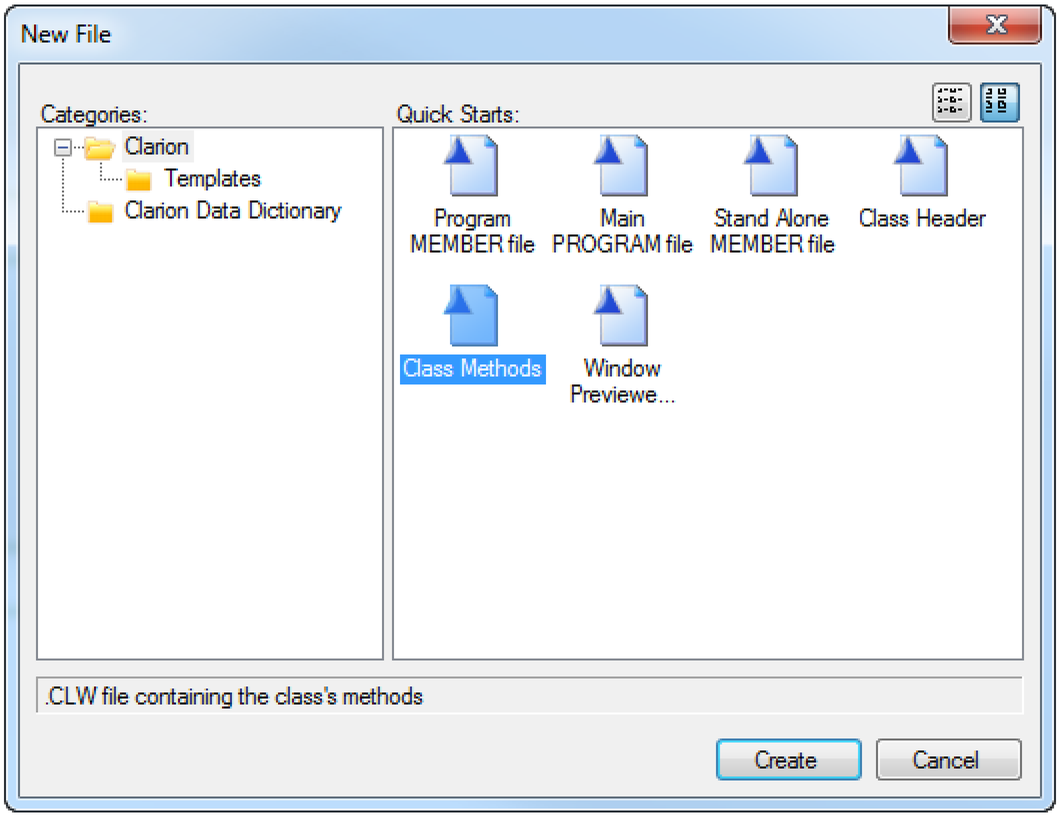


.INC file containing the class definition

And here's the resulting header:

```
MEMBER()
OMIT('***')
 * User: Dave
 * Date: 27/06/2013
 * Time: 9:24 AM
 ***
!--------------------------------------------------------------------------------
! Change ClassName? to the name of your class
!--------------------------------------------------------------------------------
!--------------------------------------------------------------------------------
! Use of the ,Type attribute is recommended, although it means you'll
! have to instantiate the class before you can use it
!--------------------------------------------------------------------------------

ClassName?     class,module('ClassName?.clw'),link('ClassName?',1),Thread!,type
!ExampleMethod              procedure
      end
```

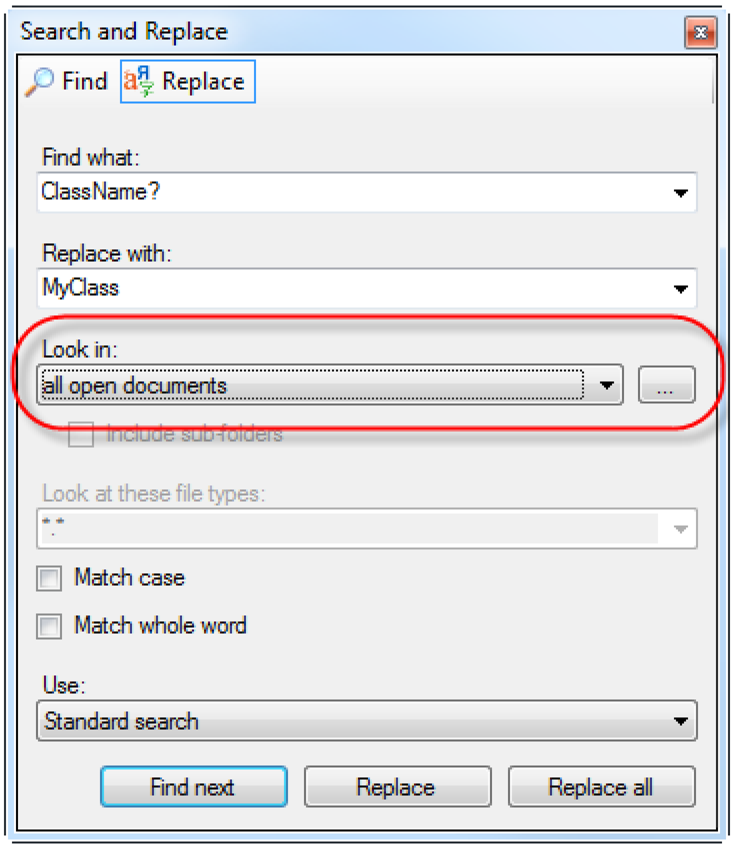
Similarly, the methods file:



And the resulting source:

```
OMIT('***')
 * User: Dave
 * Date: 27/06/2013
 * Time: 9:25 AM
 ***
!--------------------------------------------------------------------------
! Change ClassName? to the name of your class
!--------------------------------------------------------------------------

                                            Member()
                                            Map
                                            End

    include('ClassName?.inc'),once

!--------------------------------------------------------------------------
! Implement your class methods here
!--------------------------------------------------------------------------
!ClassName?.ExampleMethod        procedure
!  code
!   ! do whatever
```

Now replace ClassName? with your own class name, and do it for both files at the same time by setting Look in to all open documents.


Search and Replace

Find what: ClassName?

Replace with: MyClass

Look in: all open documents

Include sub-folders

Look at these file types: *.*

Match case

Match whole word

Use: Standard search

Save the files, specifying a file name that matches the class name. This is really important - I strongly recommend one and only one class per file, with a descriptive unambiguous name. I also use pseudo namespaces; for an example see the source for The DevRoadmaps Clarion Library (DCL).

You will have to choose a file name when you save, as the default is ClassName?.<extension> and ? is not a valid character for file names. If you're unable to save the file you probably haven't removed the ?.

The class files

I've called my class "MyClass". Here's MyClass.inc:

```
OMIT('***')
 * User: Dave
 * Date: 27/06/2013
 * Time: 10:25 AM
 ***
!---------------------------------------------------------------------------------
! Change MyClass to the name of your class
!---------------------------------------------------------------------------------
!---------------------------------------------------------------------------------
! Use of the ,Type attribute is recommended, although it means you'll
! have to instantiate the class before you can use it
!---------------------------------------------------------------------------------

MyClass     class,module('MyClass.clw'),link('MyClass',1),Thread!,type
!ExampleMethod              procedure
      end
```

And MyClass.clw

```
OMIT('***')
 * User: Dave
 * Date: 27/06/2013
 * Time: 10:25 AM
 ***
!----------------------------------------------------------------------------------
! Change MyClass to the name of your class
!----------------------------------------------------------------------------------

                                            Member()
                                            Map
                                            End

    include('MyClass.inc'),once

!----------------------------------------------------------------------------------
! Implement your class methods here
!----------------------------------------------------------------------------------
!MyClass.ExampleMethod        procedure
!  code
!  ! do whatever
```

Using the class

To use the class copy the Include statement from the .CLW file and paste it in an appropriate data embed in the program (global, module, or procedural):

include('MyClass.inc'),once

You can then call the class's methods.

To Type or not to Type? That is the question

If you look at the class header you'll see that there's a commented out Type attribute at the end of the Class line. If Type is not present then the class is automatically instantiated and ready to use. Reluctantly, I've made this the default for the quick start so that no additional steps are needed before the class can be used.

As noted in the source, however, I strongly recommend the use of ,Type. This requires you to create an instance of the class, either in an additional declaration or at runtime, but it offers much more flexibility and helps ensure you're not reusing an existing instance when you may not want to do that.

Getting the QuickStarts

The Quick Start files are included in The DevRoadmaps Clarion Library (DCL). Just grab the latest update, or download the Quck Start files directly via GitHub.

Tips & Techniques - Clarion IDE

- Adding mouse wheel support to the C6 IDE
- A fix for an IDE crash when generating with ClarionCL.exe
- ClarionCL command line parameters
- Debugging the IDE
- Dictionary Editor
- Easy OMITs
- Using a different IDE start page
- Using the task list for bookmarks in embeds
- WSLDial errors

Adding mouse wheel support to the C6 IDE

The C6 IDE lacks support for the mouse wheel, but you can add that support with one of several third party products.

The one most often used by Clarion devs is Flywheel, which you can download from any number of web sites including PC World. Another is Free wheel.

Flywheel is no longer supported but it does require registration. All users can register under the name "I Am Free" with the registration number 13601409.

A fix for an IDE crash when generating with ClarionCL.exe

I recently encountered the following IDE crash when trying to generate a C8 application via the command line using ClarionCl.exe (which you can find in your Clarion bin directory):

```
Unhandled Exception: System.NullReferenceException: Object reference not set to an
instance of an object.
   at ICSharpCode.SharpDevelop.FileService.GetOpenFile(String fileName)
   at ICSharpCode.SharpDevelop.FileService.OpenFile(String fileName)
   at
ICSharpCode.SharpDevelop.Project.ProjectService.ParserServiceCreatedProjectContents()
   at ICSharpCode.SharpDevelop.Gui.WorkbenchSingleton.SafeThreadAsyncCall(Action
method)
   at ICSharpCode.SharpDevelop.ParserService.LoadSolutionProjectsInternal()
   at ICSharpCode.SharpDevelop.ParserService.LoadSolutionProjects()
   at System.Threading.ThreadHelper.ThreadStart_Context(Object state)
   at System.Threading.ExecutionContext.Run(ExecutionContext executionContext,
ContextCallback callback, Object state)
   at System.Threading.ThreadHelper.ThreadStart()
```

The stack trace suggested a problem with LoadSolutionProjectsInternal() - could there be some bad project data? But no, I couldn't find any reason in the project data for the problem.

I'd also been making some modifications to the Clarion template chain (I'm working with a lot of legacy apps these days) and at first I thought this had something to do with a template mod gone bad. But oddly most of the apps I tried to generate worked fine; only a few crashed. And they were apps I'd worked on recently, and I'd selected some global options in those apps. I thought maybe that was the reason.

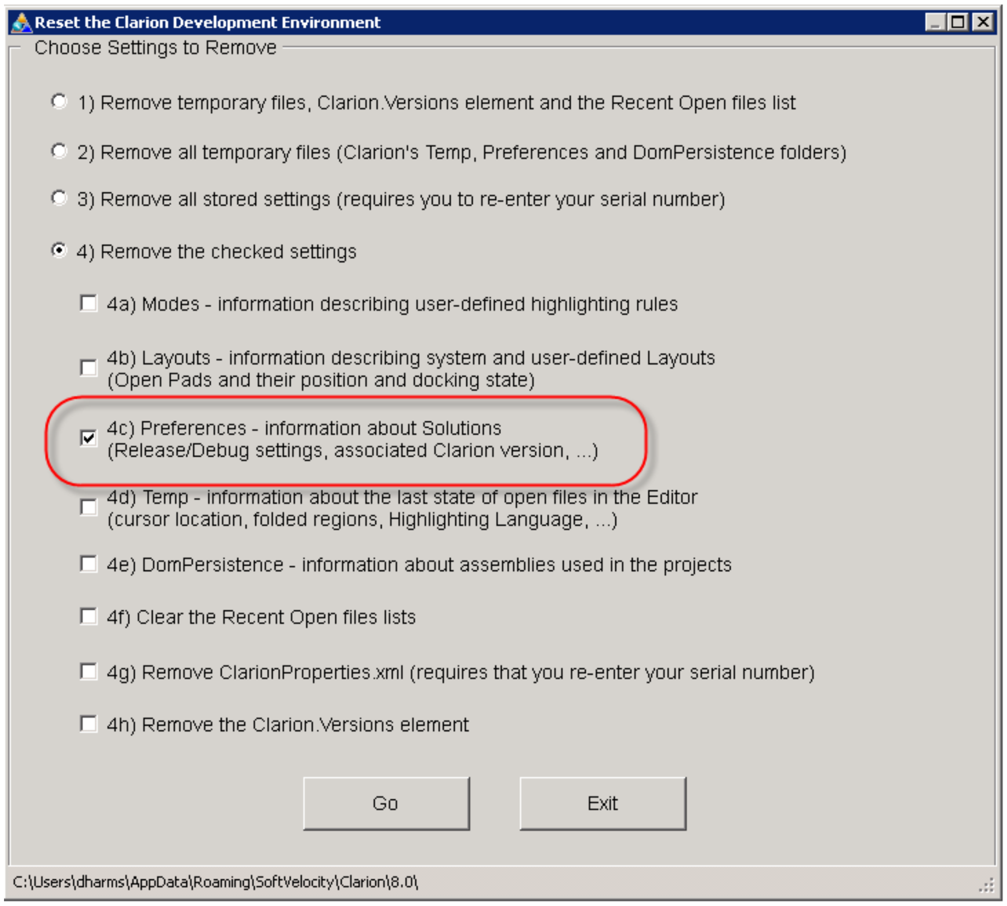
But I couldn't find a pattern; I tried copying in saved versions of the apps; the same ones crashed as before.

Finally, in desperation I renamed my %appdata%\Roaming\SoftVelocity\Clarion\8.0\Preferences folder.

Presto! ClarionCL could generate my apps again.

Something in the app preferences clearly upset Clarion's digestion.

You can also clear out preferences via the ResetIDE program that ships with Clarion.



Choose Settings to Remove

( ) 1) Remove temporary files, Clarion.Versions element and the Recent Open files list

( ) 2) Remove all temporary files (Clarion's Temp, Preferences and DomPersistence folders)

( ) 3) Remove all stored settings (requires you to re-enter your serial number)

(●) 4) Remove the checked settings

- [ ] 4a) Modes - information describing user-defined highlighting rules
- [ ] 4b) Layouts - information describing system and user-defined Layouts (Open Pads and their position and docking state)
- [✓] 4c) Preferences - information about Solutions (Release/Debug settings, associated Clarion version, ...)
- [ ] 4d) Temp - information about the last state of open files in the Editor (cursor location, folded regions, Highlighting Language, ...)
- [ ] 4e) DomPersistence - information about assemblies used in the projects
- [ ] 4f) Clear the Recent Open files lists
- [ ] 4g) Remove ClarionProperties.xml (requires that you re-enter your serial number)
- [ ] 4h) Remove the Clarion.Versions element

C:\Users\dharms\AppData\Roaming\SoftVelocity\Clarion\8.0\

ClarionCL command line parameters

The Help file lists a number of parameters for the ClarionCL command line utility. Here's one that appears to be undocumented, and was passed along to me by Ilka Ferrett.

The /p parameter lets you pass compiler settings, which in this example will set debug info (vid) to min and turn on line numbers.

```
/p:vid=min;line_numbers=True
```

If you have any other ClarionCL tips, please post them here!

Debugging the IDE

Brahn Partridge points out that you can use SharpDevelop's ILSpy to step through the decompiled source of Clarion8. You can even set breakpoints.

The SharpDevelop team began work on ILSpy after Red Gate announced that it would stop releasing thee free version of .NET Reflector after Feb 2011, and only paid versions would be available.

You can also view the IDE's XLog files using Jeff Slarve's utility, available from the downloads page.

Dictionary Editor

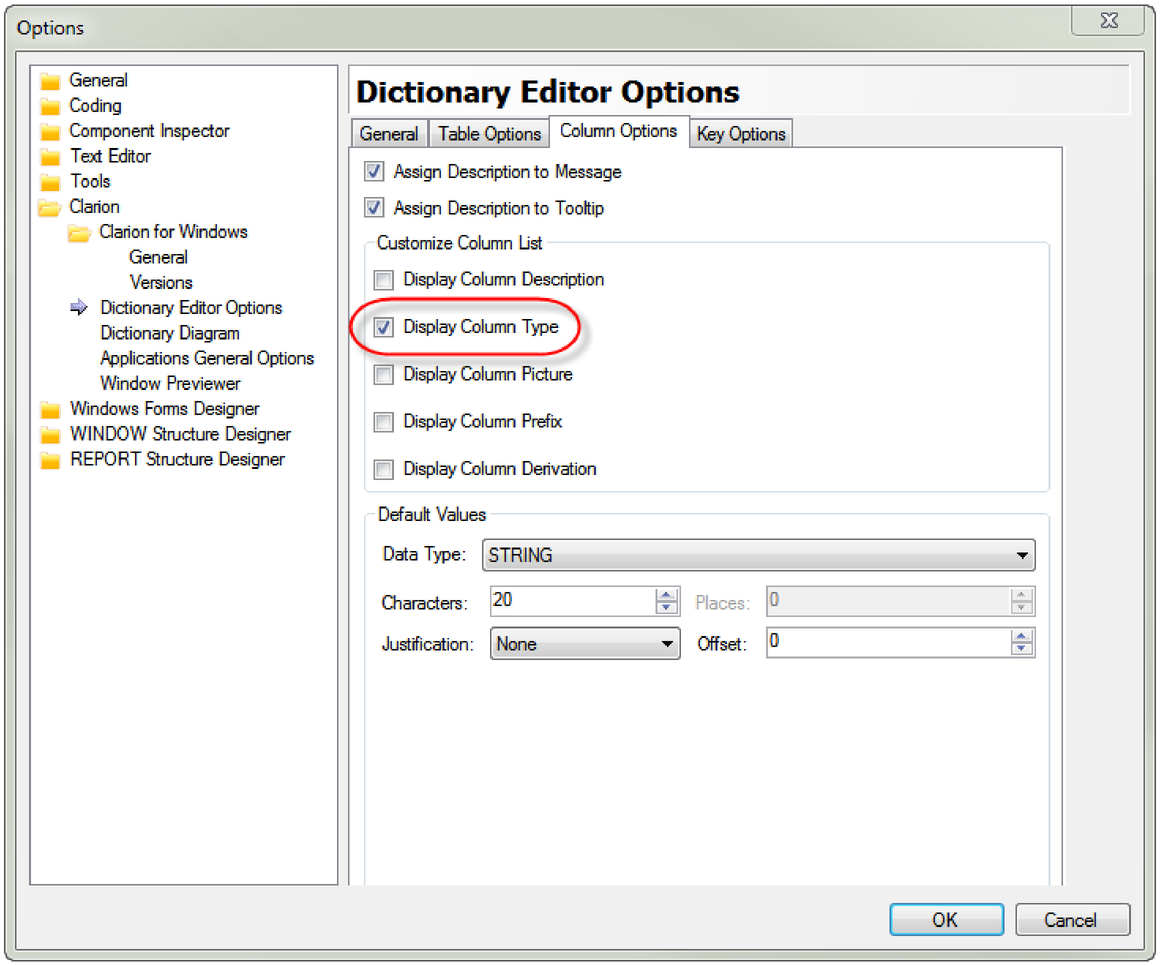
- Show the data type
- Turn off the partial save warning

Show the data type

Recently I had to go through a bunch of tables and correct some data types.

The following image isn't the dictionary in question, but it does show the view I had, which was just the column names. I had to click on each column to see the data type, and that was a pain.

Fortunately there's a way to fix this. Go to Tools | Options | Clarion | Dictionary Editor Options. Click on the Column Options tab.



Dictionary Editor Options

- ✓ Assign Description to Message
- ✓ Assign Description to Tooltip

Customize Column List

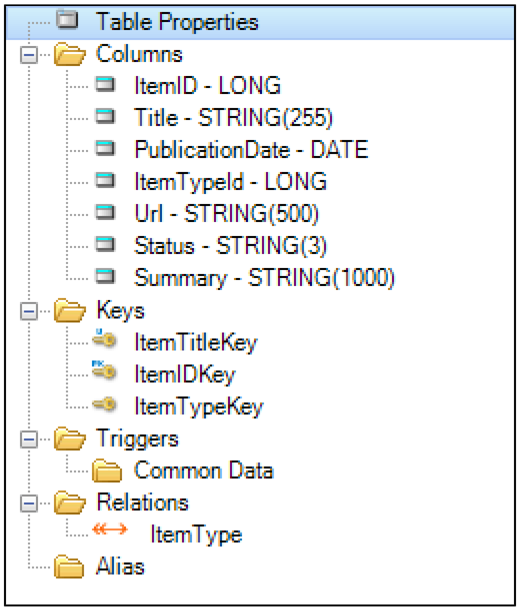
- ☐ Display Column Description
- ✓ Display Column Type
- ☐ Display Column Picture
- ☐ Display Column Prefix
- ☐ Display Column Derivation

Default Values

Data Type:	STRING		
Characters:	20	Places:	0
Justification:	None	Offset:	0

There are a bunch of things you can add, but the one I really cared about was Column Type.

That let me see the data type along with the column name, which made it a whole lot easier to see what I'd done and what still needed to be done.

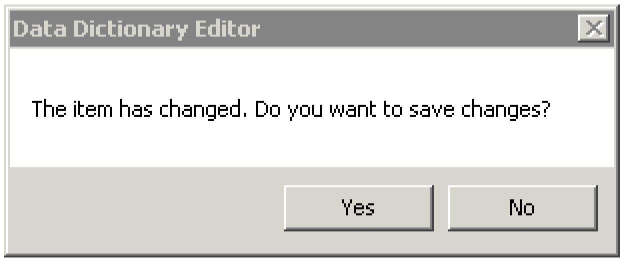


Now, if the display were columnar instead of just concatenated text that would be *really* cool....

The downside, of course, is that you don't get a chance to reject your changes to a field. But any information in the columns list isn't updated until you accept your changes (say, by clicking on another field) so if you need to back out you can always use the column display as a reference.

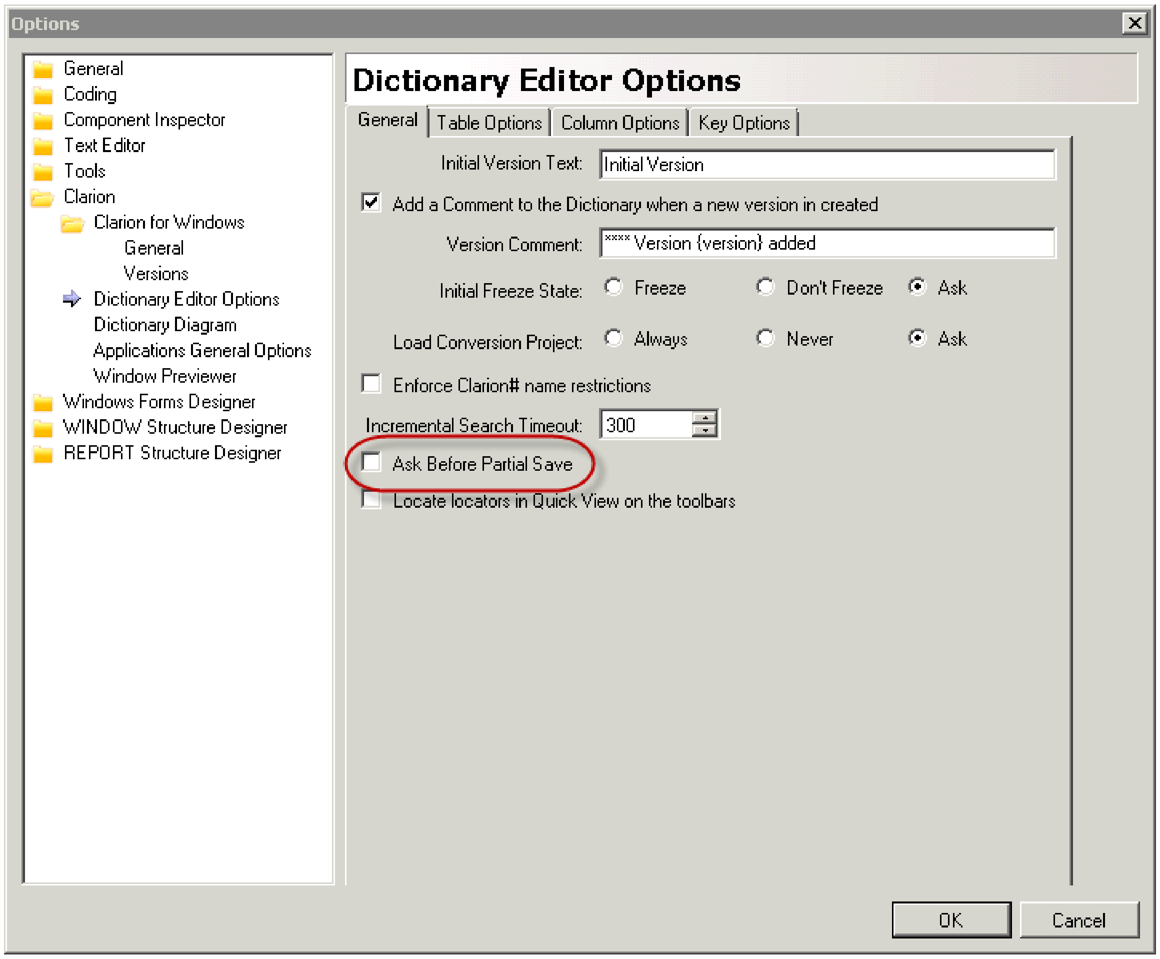
Turn off the partial save warning

You're editing a dictionary, and every time you make a field change you get the annoying warning "The item has changed. Do you want to save the changes?"


Data Dictionary Editor

The item has changed. Do you want to save changes?

The fix is easy. Go to Tools | Options and locate the Dictionary Editor Options settings for Clarion. The Ask Before Partial Save option controls the display of the message.


Options

- General
- Coding
- Component Inspector
- Text Editor
- Tools
- Clarion
  - Clarion for Windows
    - General
    - Versions
  - Dictionary Editor Options
  - Dictionary Diagram
  - Applications General Options
  - Window Previewer
- Windows Forms Designer
- WINDOW Structure Designer
- REPORT Structure Designer

Dictionary Editor Options

Initial Version Text: Initial Version

[✓] Add a Comment to the Dictionary when a new version in created

Version Comment: **** Version {version} added

Initial Freeze State: Freeze Don't Freeze [✓ Ask]

Load Conversion Project: Always Never [✓ Ask]

[ ] Enforce Clarion# name restrictions

Incremental Search Timeout: 300

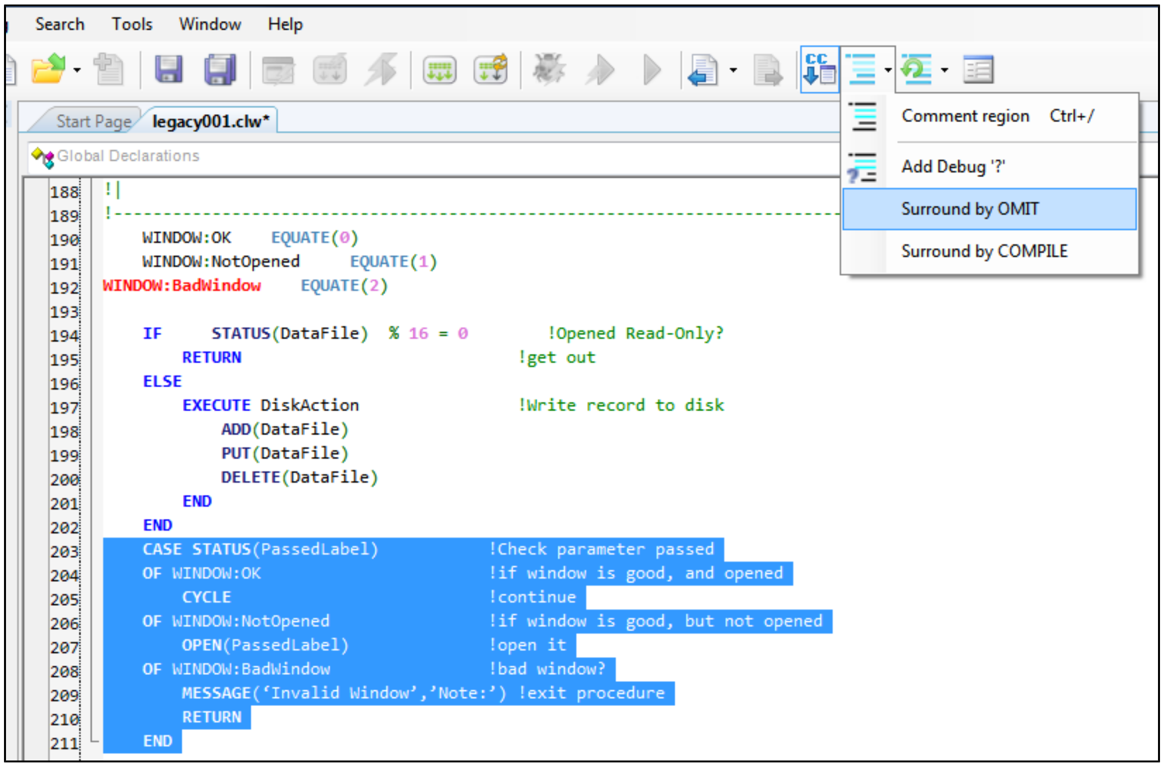
[ ] Ask Before Partial Save

[ ] Locate locators in Quick View on the toolbars

Uncheck the option and the warning message goes away.

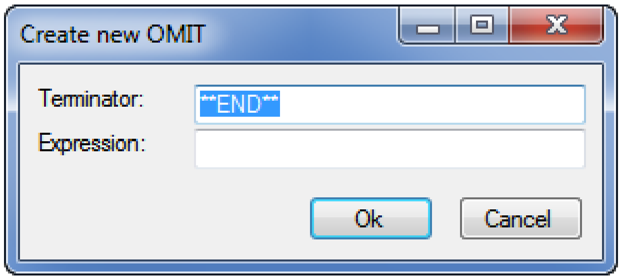
Easy OMITs

Mark Goldberg had a nice tip in the March 16 ClarionLive webinar. Highlight some text you want to omit and click on the drop button as shown:



```
!|
!-----------------------------------------------------------------------------
    WINDOW:OK     EQUATE(0)
    WINDOW:NotOpened       EQUATE(1)
WINDOW:BadWindow     EQUATE(2)

    IF     STATUS(DataFile)  % 16 = 0          !Opened Read-Only?
        RETURN                              !get out
    ELSE
        EXECUTE DiskAction                  !Write record to disk
            ADD(DataFile)
            PUT(DataFile)
            DELETE(DataFile)
        END
    END
    CASE STATUS(PassedLabel)               !Check parameter passed
    OF WINDOW:OK                           !if window is good, and opened
        CYCLE                              !continue
    OF WINDOW:NotOpened                    !if window is good, but not opened
        OPEN(PassedLabel)                  !open it
    OF WINDOW:BadWindow                    !bad window?
        MESSAGE('Invalid Window','Note:') !exit procedure
        RETURN
    END
```

You'll get another window where you can type in the terminator and an optional expression:



With no expression you'll get this:

```
OMIT('**END**')
    CASE STATUS(PassedLabel)           !Check parameter passed
    OF WINDOW:OK                       !if window is good, and opened
        CYCLE                          !continue
    OF WINDOW:NotOpened                !if window is good, but not opened
        OPEN(PassedLabel)              !open it
    OF WINDOW:BadWindow                !bad window?
        MESSAGE('Invalid Window','Note:') !exit procedure
        RETURN
    END
    !end of OMIT('**END**')
```

Type in an expression or value (you'll need to define an equate somewhere to define the value) and you get a conditional omit. With an expression of MyFlag you get this:

```
OMIT('**END**', MyFlag)
    CASE STATUS(PassedLabel)           !Check parameter passed
    OF WINDOW:OK                       !if window is good, and opened
        CYCLE                          !continue
    OF WINDOW:NotOpened                !if window is good, but not opened
        OPEN(PassedLabel)              !open it
    OF WINDOW:BadWindow                !bad window?
        MESSAGE('Invalid Window','Note:') !exit procedure
        RETURN
    END
    !end of OMIT('**END**', MyFlag)
```

Using a different IDE start page

I do a lot of consulting on a massive Clarion project made up of a few hundred APP files. Some time ago I set up continuous integration using TeamCity and Git (which is another story I'd like to get to in the near future) mainly so I could do automated builds on nightly checkins into the Git repository.

Naturally, I want to know the status of those builds. I use TeamCity's Window Tray Notifier, a little app that sits in the system tray and gives me a visual indicator of the status of the four or five projects (individual releases of that massive app) that may be building on any given night.

Only lately I can't seem to get get the Tray Notifier to report on all those apps - it just seems to want to report on two of them. I should find out why.

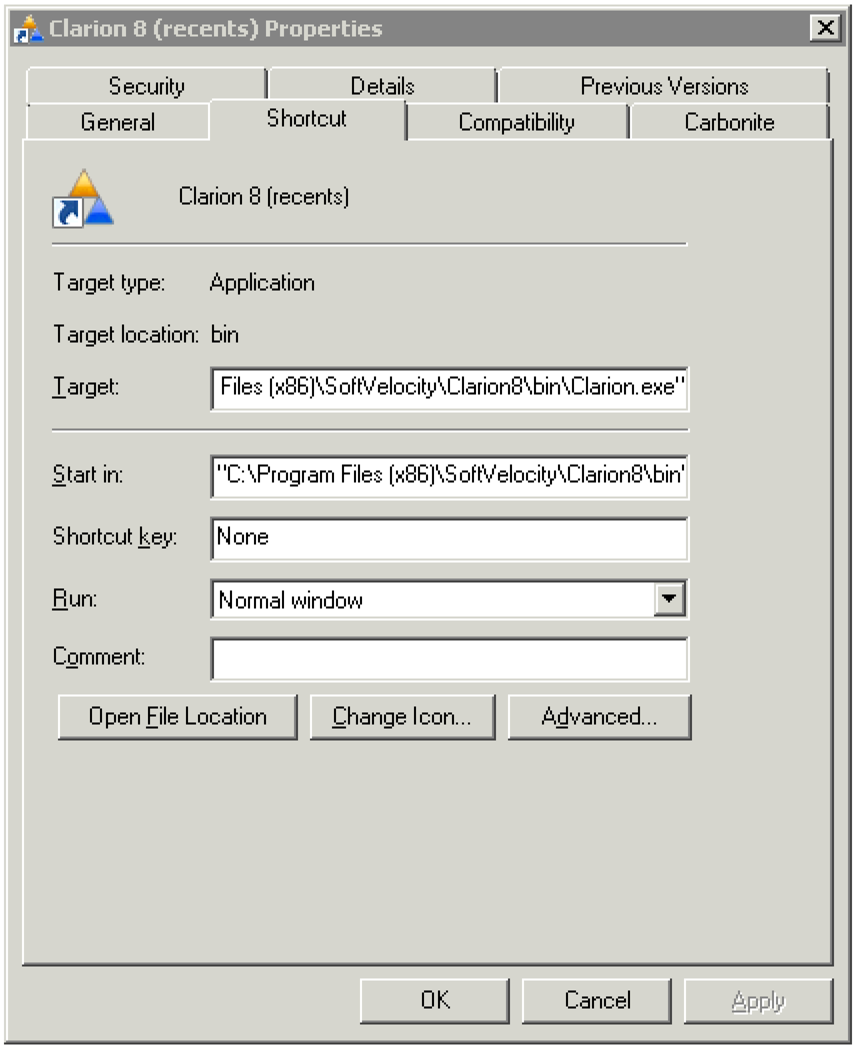
But in the meantime I had another thought. TeamCity uses a web interface, so when I want to see all of the currently active projects I'm actually just launching a browser.

And doesn't the IDE use an embedded browser for its start page?

So I got to wondering if there was some way to configure the IDE to show a specific web page instead of its own start page. I didn't find a way to do that in the IDE's configuration files (if you know how, please tell me) but I did find out that if you pass a URL as a parameter to Clarion.EXE the IDE will display that URL. Problem solved.

The only minor hassle is that to display the regular start page I have to choose View | Start Page (or press Alt-V,S).

So now I have two Clarion8 shortcuts. I've named one Clarion 8 (recents):



Target type: Application

Target location: bin

Target: Files (x86)\SoftVelocity\Clarion8\bin\Clarion.exe"

Start in: "C:\Program Files (x86)\SoftVelocity\Clarion8\bin"

Shortcut key: None

Run: Normal window

Comment:

and it has the usual Target value:

```
"C:\Program Files (x86)\SoftVelocity\Clarion8\bin\Clarion.exe"
```

The other shortcut is called Clarion 8 (TeamCity) and it has this target:

```
"C:\Program Files (x86)\SoftVelocity\Clarion8\bin\Clarion.exe"
http://localhost:8111/overview.html
```

Since I'm mainly concerned with the status of the nightly builds, I'll usually run the TeamCity version once in the morning; after that I'll stick with the regular shortcut.

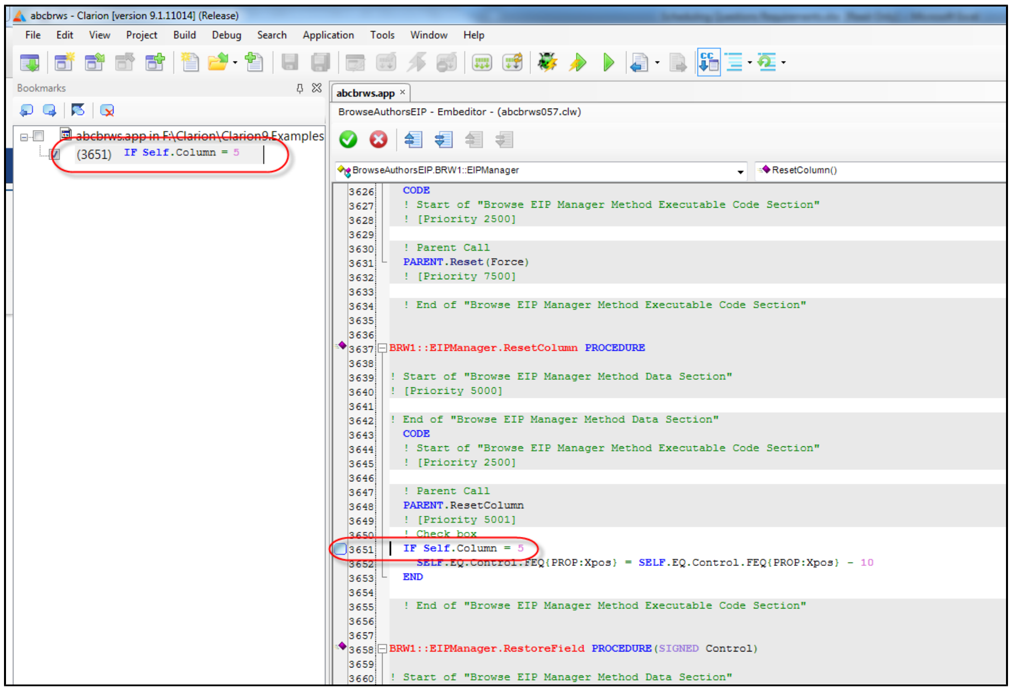
Using the task list for bookmarks in embeds

Although I spend a lot of my Clarion programming time writing classes and unit tests for those classes, I also spend a certain amount of time in regular Clarion procedures, navigating back and forth between embeds (in the embeditor, naturally).

The more embeds I'm using, the more jumping around I tend to do. If the embeds are close together it's no big problem. If they're all over the procedure it's more of a hassle to quickly get to where I need to be.

The Clarion IDE does have a nifty bookmarking capability - just press Ctrl-F2 on any line and a bookmark appears in the bookmarks list. (Ctrl-F2 again to remove the bookmark, or click on the ilttle button icon that appears to the left of the line number.) Double click on the bookmark to go to that line in the source; if the file isn't open the IDE will open it for you and take you to that line. Bookmarks are persistent for source files, so they'll be there the next time you load that project.

But what about APPs? It turns out that bookmarks work within the embeditor:


abcbrws - Clarion [version 9.1.11014] (Release)
File Edit View Project Build Debug Search Application Tools Window Help
Bookmarks
abcbrws.app
BrowseAuthorsEIP - Embeditor - (abcbrws057.clw)
BrowseAuthorsEIP.BRW1::EIPManager
ResetColumn()

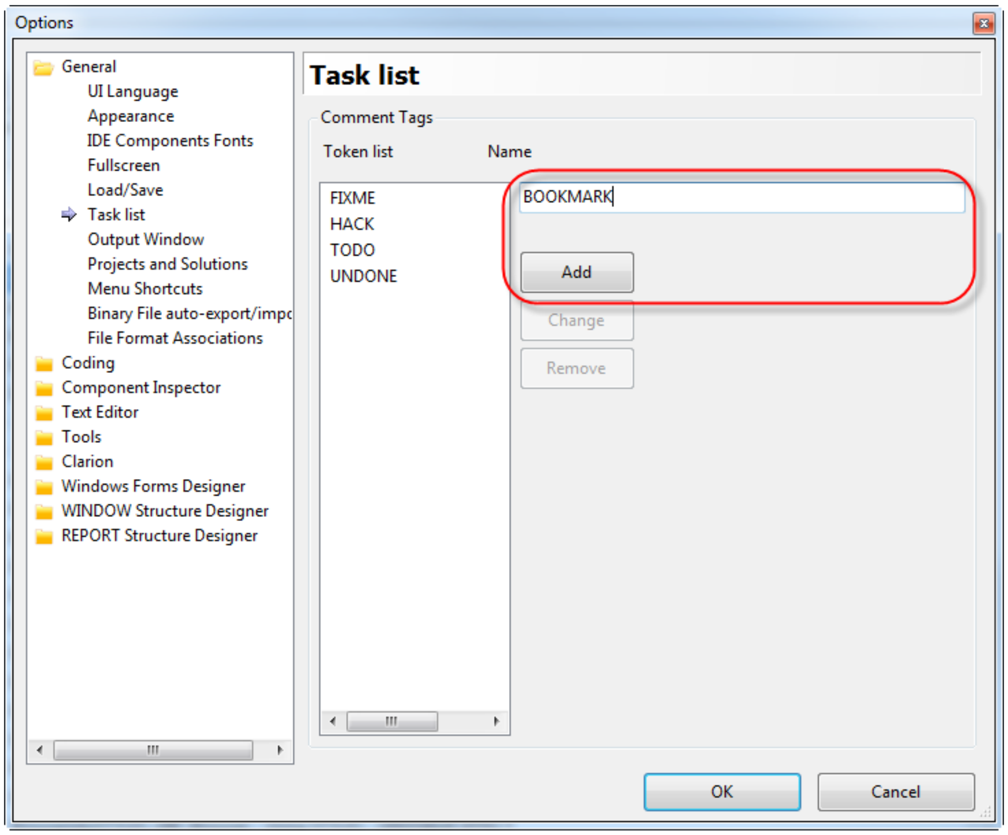
```
abcbrws.app in F:\Clarion\Clarion9.Examples
  (3651)  IF Self.Column = 5
```

```
   CODE
   ! Start of "Browse EIP Manager Method Executable Code Section"
   ! [Priority 2500]

   ! Parent Call
   PARENT.Reset(Force)
   ! [Priority 7500]

   ! End of "Browse EIP Manager Method Executable Code Section"


BRW1::EIPManager.ResetColumn PROCEDURE

! Start of "Browse EIP Manager Method Data Section"
! [Priority 5000]

! End of "Browse EIP Manager Method Data Section"
   CODE
   ! Start of "Browse EIP Manager Method Executable Code Section"
   ! [Priority 2500]

   ! Parent Call
   PARENT.ResetColumn
   ! [Priority 5001]
   ! Check box
   IF Self.Column = 5
     SELF.EQ.Control.FEQ{PROP:Xpos} = SELF.EQ.Control.FEQ{PROP:Xpos} - 10
   END

   ! End of "Browse EIP Manager Method Executable Code Section"


BRW1::EIPManager.RestoreField PROCEDURE(SIGNED Control)

! Start of "Browse EIP Manager Method Data Section"
```

But as soon as you close the embeditor any bookmarks you've created in that procedure disappear from the bookmarks view.

A bookmarking workaround for apps

There is however an easy way to create persistent bookmarks for embed points. Go to Tools | Options | Task List and add a token called BOOKMARK (or any other text you like):



Anywhere you want a bookmark in your code, put the text

!BOOKMARK

followed by whatever text you want to appear in the bookmark. You might add something descriptive such as a routine name:

!BOOKMARK BRW1::EIPManager.SaveOnChangeAction

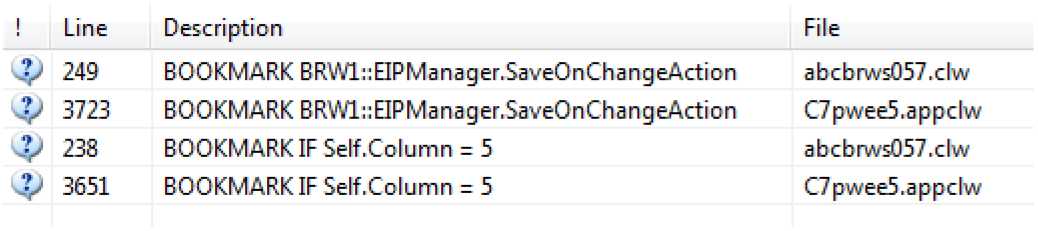
With the embeditor open, view the task list and look for lines that begin with BOOKMARK. Double-click on any item to jump to that location in the code.

The downside

This is a neat trick, but it does have couple of issues.

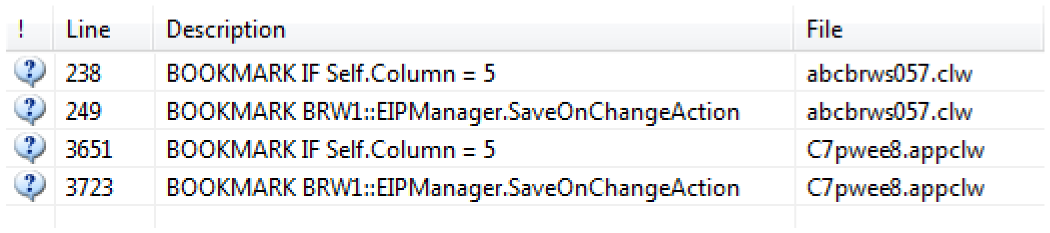
For one, the bookmarks only show up when you're in the embeditor.

For another, once the code is generated you may find that you have duplicated line entries:



!	Line	Description	File
?	249	BOOKMARK BRW1::EIPManager.SaveOnChangeAction	abcbrws057.clw
?	3723	BOOKMARK BRW1::EIPManager.SaveOnChangeAction	C7pwee5.appclw
?	238	BOOKMARK IF Self.Column = 5	abcbrws057.clw
?	3651	BOOKMARK IF Self.Column = 5	C7pwee5.appclw

This is because the IDE is showing you both the embeds in the embeditor and in the generated source. To avoid confusion click on the File header to sort the list in file order - that way you'll at least have all of the embeditor bookmarks in the same location.



!	Line	Description	File
?	238	BOOKMARK IF Self.Column = 5	abcbrws057.clw
?	249	BOOKMARK BRW1::EIPManager.SaveOnChangeAction	abcbrws057.clw
?	3651	BOOKMARK IF Self.Column = 5	C7pwee8.appclw
?	3723	BOOKMARK BRW1::EIPManager.SaveOnChangeAction	C7pwee8.appclw

There's another issue, which is that unlike the bookmarks list, which a tree view, this is a straight list of tasks and you can't hide tasks from other files by collapsing a tree node.

And finally this list contains all tasks - if you have !TODO entries in your code they'll show up along with your bookmkarks.

That said, this is still a useful trick and can be a real timesaver if you're spending a lot of time in a procedure and you need to regularly move between distant embed points.

WSLDial errors

WSLDial errors were more common in the early days of Clarion for Windows, but they still crop up every now and then.

There are four reported WSLDial errors, numbers 01 through 04.

WSLDial 01

This error

Knowledge Base Reference: 100023   Dated: 16/07/2001

Clarion WSLDIAL 01, 02, 03 or 04 errors

These are internal program errors.
Please notify dka ASAP if you are getting these errors and specify
exactly where the error occurs.

Note to programmers

WSLDIAL 01 error could be:
- A screen or report variable is missing a ? in front of the variable.
- You are using a reserved word as a variable

WSLDIAL 02 error for which the program is saying:
"I can't find a resource that I need"
eg. it can't find a window, icon or bitmap resource.

- Do Module names exceed 8.3 format and length?
- Can all icons be located.
- Delete all clw/obj/rsc files for dll's/lib's/exe and recompile all.

WSLDIAL 03 error
- A procedure is calling itself.

WSLDIAL 04 error
This error is generated by incorrect data definitions.
Declarations must be the same throughout apps and dlls
eg: SampleQ Queue,THREAD,external,dll
SampleQ Queue,THREAD,dll   (Missing the external attribute)

Understanding Clarion's build terminology