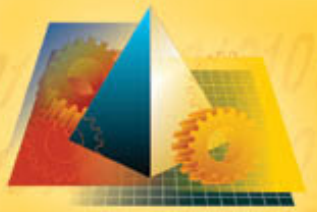


Clarion

IDE Reference

SoftVelocity Incorporated

2335 East Atlantic Blvd. Suite 410

Pompano Beach, Florida 33062

(954) 785-4555

www.softvelocity.com

Documentation for Clarion 7

Printed in the United States of America (0409)

Table Of Contents

IDE Reference

What's New in Clarion 7

Here is a categorized listing of some of the many new features available in this version.

IDE

Clarion 7 debuts the first 32-bit IDE, in part using the critically acclaimed Sharp Develop (#Develop) technology. However, this IDE is unique and special to Clarion, particularly in the area of rapid application development with a new Dictionary Editor and Application Generator.

With Clarion 7, multiple instances of the IDE can now be opened

- A New Integrated Development Environment (IDE) based on .Net Framework 2.0 and higher.

Dictionary Editor

- Powerful grouping features to allow 'subset views' of the tables in the Dictionary
- Display Graphical views of tables and relationships with the new Dictionary Diagrammer
- Scan all important elements of your active dictionary using the new Diagrammer Reports View
- A new column view that displays every field from all tables and allows global edits
- Cascade any field change throughout the entire Dictionary
- Improved SQL Scheme generation
- Open multiple tables for simultaneous editing
- Arrange your own personal layout for editing of Fields, Keys and Relationships
- Powerful new List view with Edit-in-place that allows you to specify which Dictionary attributes to display
- Validity check option now supports "Allow Null"

Application Generator

- Completely re-factored for blazing code-generation speed!
- Improved views of Procedures and Classes
- Simultaneously Open multiple app files

Project System

- Create 'Solutions' which can contain any number of individual Projects. This allows you to build any number of Executables and DLLs from a single solution
- Build both Debug and Release configurations from a single solution
- Ability to build any Project with any installed version of Clarion back to Clarion 4, using the corresponding versions' compiler, templates, and include files specified in the Redirection file.
- Ability to pre-process any project with local or global information
- Add extra tasks to any Clarion project via the MSBuild system,(the new and extensible, XML-based build engine that is included with the .NET Framework 2.0).

Enhanced Database Drivers

- TopSpeed driver has new Encryption support. Use any encryption algorithm, supported by any encryption provider that plugs into the Windows encryption subsystem; including the *Microsoft Base Cryptographic Provider* and *Microsoft Enhanced Cryptographic Provider.*
- SQL driver(s): many enhancements designed to improve flexibility and performance.

New Source Editor

New productivity features such as code folding, access to the Dictionary from within hand-coded projects, Code Regions, and Procedure and Class navigators

Templates

- Powerful "Toolbox" support
- Built-in extended Vista manifest support
- New **ClarionCL** executable included to allow you to register templates outside of the IDE (replaces outdated DDE support).

New Template Editor

- Editing your template files has never been easier. New productivity features such as code folding, quick section searches in drop list, and fast symbols and declaration searches in the context menu

Brand-new Report Designer and Window Designer with many productivity enhancements.

User Interface (UI) improvements, including:

- XP Office style Menu's
- Tabbed MDI windows
- Four new Graphical styles for Sheet/Tabs
- Complete XP Theme support for all controls
- RichEdit support upgraded
- Support for embedded Tables and URLs

Language Reference:

- FILE structures can now be TYPE'd. The importance of this is significant, where it is now possible to have two separate record buffers on the same thread.

PROP:ThemeActive can now be used to test a specific control.

New SYSTEM properties for hooks for OPEN and CLOSE windows and reports
PROP:OpenWindowHook
PROP:CloseWindowHook
PROP:OpenReportHook
PROP:CloseReportHook

Template Language Reference:

- Addition of new EDIT and ICON type attribute for #PROMPT
- New SPLIT and UNSPLIT built-in template procedures
- Extended library support for the #PROJECT statement

Extended UI Template Support for Clarion 7

As an added feature to the initial beta release of Clarion 7, Clarion 6 includes a special template interface designed to add powerful XP styles easily to your existing applications that you are planning to migrate to Clarion 7.

Projects in the initial release of Clarion 7 are first exported to an external project file from a Clarion 6 application, which is then imported and processed by the Clarion 7 IDE. Using this technique, the Extended UI interface is used to include special styles that Clarion 7 supports from inside your current Clarion 6 application. This includes XP Style Menus, MDI tab controls, and special enhancements for SHEET and TAB controls.

This Extended UI support will not affect your existing Clarion 6 applications. A special compiler flag detects that the code generated to support the styles you implement will only be incorporated into your Clarion 7 programs.

And of course, this support is also available for all new applications that you create in Clarion 7.

Extended UI support is provided in the following areas, for both ABC and Clarion template chains:

Frame Properties

SHEET Control Properties

TAB Control Properties

Click on the links above for more detailed information regarding this support.

Enhanced Encryption Support in the TopSpeed driver

The TopSpeed driver already has a very secure encryption system. However, it is not based on any of the standard encryption algorithms. This makes the TopSpeed file format unavailable to some developers who have to guarantee the encryption algorithm. With the introduction of Clarion 7.0, you will have the ability to use any encryption algorithm supported by *any* encryption provider that plugs into the Windows encryption subsystem. This will enable developers to create, store and exchange data in a very secure environment.

There are two providers that are installed on all Windows Operating Systems: ***Microsoft Base Cryptographic Provider*** and ***Microsoft Enhanced Cryptographic Provider***.

The Microsoft Enhanced Cryptographic Provider supports the same capabilities as the Microsoft Base Cryptographic Provider, but provides for stronger security through longer keys and additional algorithms. The Enhanced provider is installed on your machine when you apply the Internet Explorer 128-bit security patch.

In addition, you now have the ability to use any encryption algorithm supported by *any* encryption provider that plugs into the Windows encryption subsystem. So you can now even use *new* encryption providers as they become available!

To enable use of an alternative encryption algorithm you just supply at least one of the following encryption settings to the driver via the driver string:

Example Format:

(DriverString = Default Value)

/PROVIDER= PROV_RSA_FULL 1

Driver String	Default Value	Description
/PROVIDER	Microsoft Enhanced Cryptographic Provider	The name of the cryptographic service provider
/CONTAINER	NULL	The container within the provider where the key algorithm is located (see (2) below)
/PROVIDERTYPE	(Full RSA)	The encryption type (see (3) below)
/KEYALGORITHM	(RC4)	The algorithm used to encrypt data
/HASHALGORITH	(MD5)	The algorithm used to hash the password (see (4) below)
/FORCEKEY	FALSE	Set to FALSE to use the default key algorithm if the supplied key algorithm is not available or not valid
/FORCEHASH	FALSE	Set to FALSE to use the default hash algorithm if the supplied hash algorithm is not available

Additional Notes:

(1) You can use any of the driver string switches in a SEND command before the file is open to set the encryption algorithm. You can also retrieve the current value of any of the encryption options as the return result of the SEND command.

(2) **/CONTAINER** is the key container name. This is a string that identifies the key container to the CSP (cryptographic service provider). This name is independent of the method used to store the keys. Some CSPs store their key containers internally (in hardware), some use the system registry, and others use the file system.

When /CONTAINER is not specified, a default key container name is used. For example, the Microsoft Base Cryptographic Provider uses the logon name of the currently logged on user as the key container name. Other CSPs can also have default key containers that can be acquired in this way.

(3) This is the value associated with the different provider types. See the Microsoft Cryptographic Provider Types page for full details on provider types. The values of the different provider types supplied by the Microsoft Enhanced Cryptographic Provider (MECP) are:

PROV_RSA_FULL 1
PROV_RSA_SIG 2
PROV_DSS 3
PROV_FORTEZZA 4
PROV_MS_EXCHANGE 5
PROV_SSL 6
PROV_RSA_SCHANNEL 12
PROV_DSS_DH 13
PROV_EC_ECDSA_SIG 14
PROV_EC_ECNRA_SIG 15
PROV_EC_ECDSA_FULL 16
PROV_EC_ECNRA_FULL 17
PROV_DH_SCHANNEL 18
PROV_SPYRUS_LYNKS 20
PROV_RNG 21
PROV_INTEL_SEC 22
PROV_REPLACE_OWF 23
PROV_RSA_AES 24

(4) Values supported by the Microsoft Enhanced Cryptographic Provider are:

32769 MD2
32771 MD5
32772 U.S. DSA Secure Hash Algorithm
32773 Message Authentication Code
32776 SSL3 client authentication
32777 HMAC, a keyed hash algorithm

(5) Some values supported by various providers are:

17921 DES
17922 RC2
17923 3DES
17929 3DES 112
18433 RC4

For full list of key and hash algorithms the user should consult the documentation of their cryptography provider.

Multiple Version Support

The Clarion 7 IDE supports version switching for all projects and applications.

For hand coded projects, you can switch versions at any time. If you are switching versions for applications, the application must be closed prior to switching versions (due to the template registry processing that is required).

From the IDE Menu, select **Build > Set Clarion Version** to switch versions. Versions are auto-detected by the Clarion 7 IDE, but you may also manually add versions through the Tools > Options > Clarion > Clarion for Windows > Options dialog.

See: Tools Options: Clarion - Clarion for Windows - Versions

When working with applications and multiple versions, each version uses a separate registry file. When switching versions for the first time, you will have to register the templates that will be used for that version. Use the **Tools > Edit Template Registry** menu item to do this.

Here is a step-by-step guide to working with C7 applications using the C6 template registry and compiler:

1. Start the Clarion 7 IDE.
2. Switch to the C6 version (**Build > Set Clarion Version**)
3. Go to **Tools > Edit Template Registry**
4. Register the needed C6 templates.
5. Load the target C6 application (if the application has not been previously converted).
6. After conversion and loading, you should now be able to Build and Run (if an executable target) a C6 version of the application from the C7 IDE.

New Properties/Events in Clarion 7

New Properties for Clarion 7:

Support for tabbed MDI windows are now added to the Clarion 7 runtime library.

This capability is implemented by using the following new properties that are applied the application frame window:

Property	Values	Description
AppFrame{PROP:TabBarVisible}	TRUE FALSE	Turns on/off visibility of the tab bar
AppFrame{PROP:TabBarStyle}	1 – "black and white" in Office 2003 style 2 - colored in Office 2003 style 3 - squared tab 4 - boxed tab	Specifies the visual style of the tab bar
AppFrame{PROP:TabBarLocation}	0 – at the top of the frame window 1 – at the bottom of the frame window	Specifies location of the tab bar
?CurrentTab{PROP:TabSheetStyle}	0 – "black and white" in Office 2003 style 1 - colored in Office 2003 style 2 - squared tab 3 - boxed tab	Specifies the visual style of the sheet/tab control

New properties for coloring selection bar frame:

PROPLIST:BarFrame

PROPSTYLE:BarFrame

New SHEET Property:

PROP:TabRect

Returns the coordinates of a target SHEET control.

New printer properties:

PROPPRINT:SupportCopies
A READ-ONLY property that returns TRUE if the current printer supports output of multiple copies.

PROPPRINT:SupportCollate
A READ-ONLY property that returns TRUE if the current printer supports collating of copies.

New Events for Clarion 7:

EVENT:NestedLoop

This event is posted on entry to a nested ACCEPT loop

New Templates for Clarion 7

New Template Symbols added:

%LocalDataID, %LocalDataFull, %ModuleDataID, %ModuleDataFull, %GlobalDataID and %GlobalDataFull are added to make clear the differences between %GlobalData and %ModuleData/%LocalData for fields of compound structures with a prefix

Deprecated Items and Features

Several items and features have been deprecated in this release. The following detailed list presents a summary:

Properties no longer valid:

PROP:vbxfile

PROP:vbxname

PROP:vbxEvent

Lessons and Examples Location

In the Clarion 7 install, all shipping Examples and Lessons are now installed in the Common Document s area, instead of the install root folder as in prior versions.

This location can vary according to operating system.

For example, in Windows XP, the Examples and Lessons folders are found in:

C:\Documents and Settings\All Users\Documents\SoftVelocity\Clarion7\Examples

In Vista and Windows 7:

C:\Users\Public\Documents\SoftVelocity\Clarion7\Lessons

Documentation

Manuals in PDF Format

This release contains all the manuals in portable document format (PDF). You can read the PDF manuals with the latest Adobe® Reader® available from a download link on the install CD.

The PDF versions of the manuals are also indexed to allow fast searches across all manuals.

One change that you may have noticed is that printed documentation is no longer included with the product. However the complete PDF document set is available on the CD. Printed manuals are available for purchase. Please see our web site at www.softvelocity.com for more information on purchasing printed manuals.

Clarion 7 Manuals Listing

ABC Library Reference

This is the handbook to the Clarion tools that you'll use the most during your application development. It fully documents the Application Builder Class (ABC) Library used in the template-generated code. Due to its size, the printed version is split into two volumes.

Advanced Programming Resources

Contains articles on Dictionary and Application text file formats (TXD and TXA respectively).

Advanced Topics and Reference Guide

Includes documentation for the Clarion Language Utilities, Project System Reference, Multi-Language Programming Guide, and using API calls and other advanced resources.

Clarion Language Programming

This book is a collection of in-depth articles on various aspects of Clarion language programming, including program structure, object oriented programming, database design, data file processing, multi-user considerations, and developing client/server applications.

Clarion XML Support and Reference Guide

This guide explores the new classes and templates used for XML support in Clarion.

Database Drivers
This book provides in-depth information on all the file drivers available for Clarion.

FAQs, Tips and Tricks

All of the "How To" articles are included here, in addition to some general articles and recommended development and programming techniques.

IDE Reference

Provides a task-oriented description of the development environment, arranged by its major components.

Language Reference

The complete guide to the Clarion language. This book provides descriptions of all Clarion language statements, with examples for each. The Language Reference is organized by functional topics. The full text of the Language Reference is also in on-line Help. When working in the Text Editor, place the insertion point on any Clarion language statement or function, and then press the F1 key to view help for the item. Due to its expanded size, the printed version is split into two volumes.

ReportWriter User's Guide

Documents Clarion's ReportWriter for Windows--a stand-alone end-user report writing tool (also available as a separate product).

Template Language Reference
Contains the complete guide to Clarion's Template language, providing descriptions of all its statements and functions, with examples for each, clearly demonstrating how to write your own templates.

Template User's Guide
This book is a complete guide to Clarion's shipping template sets.

Getting Started

Setting Up Your Development Environment

Everything that you need to set up your development environment should be handled by the Clarion install. In general, here is summary of the elements you need:

The .NET 2.0 runtime (Clarion will install this if it is not detected)

The .NET 2.0 SDK (If you will developing with Clarion.NET)

Project System Quick Start

The Clarion 7 project system has been updated and upgraded to use the MSBuild project engine. MSBuild is the new extensible, XML-based build engine that ships with the .NET Framework 2.0 and higher. MSBuild simplifies the process of creating and maintaining build scripts, and formalizes the steps required to institute a formal build process.

The complete reference guide to MSBuild can be found at the following link:

http://msdn2.microsoft.com/en-us/library/wea2sca5(VS.80).aspx

All Clarion 7 projects use a default extension of *.CWPROJ.

Here is a sample listing:

```
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

  <PropertyGroup>

    <OutputName>people</OutputName>

    <Model>Dll</Model>

    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>

    <Platform Condition=" '$(Platform)' == '' ">Win32</Platform>

    <DefineConstants>_ABCDllMode_=&gt;0;_ABCLinkMode_=&gt;1</DefineConstants>

    <ProjectGuid>{FFFF1465-30CB-4F2D-A01C-AF2E646CA106}</ProjectGuid>

  </PropertyGroup>

  <PropertyGroup Condition=" '$(Configuration)' == 'Debug' ">

    <vid>full</vid>

    <check_stack>True</check_stack>

    <check_index>True</check_index>

  </PropertyGroup>
```

```
  <PropertyGroup Condition=" '$(Configuration)' == 'Release' ">

    <vid>off</vid>

    <check_stack>False</check_stack>

    <check_index>False</check_index>

  </PropertyGroup>

  <ItemGroup>

    <FileDriver Include="TOPSPEED" />

    <Compile Include="PEOPLBC0.CLW" />

    <Compile Include="PEOPLBC.CLW" />

    <Compile Include="people.clw" />

    <Compile Include="peopl001.clw" />

  </ItemGroup>

  <Import Project="$(ClarionBinPath)\SoftVelocity.Build.Clarion.targets" />

</Project>
```

All projects compiled in Clarion 7 are stored in solution files (*.SLN). These files can store multiple projects, and allow multiple resources to be referenced and included.

Think of the projects used in earlier versions of Clarion as a subset of any solution.

Note:

For your convenience, we also have included the project system documentation for the versions of Clarion prior to Version 7.

Hot Key Quick Reference

The IDE Menu in Clarion 7 and Clarion# is packed with many options and features. To assist you in becoming familiar with the many hot keys used, a "Quick Reference" PDF ships with this product, and is located in the install's DOCS folder.

For your convenience, the commonly used hot keys, and their corresponding function is shown below:

Hot Key	Function
ALT+F1	Content Help
ALT+F2	Previous Bookmark
ALT+F8	Rebuild Solution
ALT+SHIFT+F	Data Dictionary > Fields
ALT+SHIFT+K	Data Dictionary > Keys
CTRL+A	Select All
CTRL+ALT+B	Break
CTRL+ALT+B	Breakpoints
CTRL+ALT+C	Callstack
CTRL+ALT+F	Files
CTRL+ALT+G	Code Coverage
CTRL+ALT+G	Global Variables
CTRL+ALT+H	Threads
CTRL+ALT+K	Errors
CTRL+ALT+L	Projects
CTRL+ALT+O	Output
CTRL+ALT+T	Unit Tests
CTRL+ALT+U	Loaded Modules
CTRL+ALT+V	Local Variables
CTRL+ALT+W	Watch

CTRL+ALT+X	ToolBox
CTRL+B	Goto Matching Brace\Jump to Previous Filled Embed
CTRL+C	Copy
CTRL+D	Structure Designer
CTRL+E	Incremental Search
CTRL+F	Find
CTRL+F2	Toggle Bookmark
CTRL+F3	Find Next Selected
CTRL+F4	Close File
CTRL+F5	Start without debugger
CTRL+G	Goto Line Number
CTRL+I	Indent
CTRL+J	Jump to next filled embed
CTRL+N	New File
CTRL+O	Open
CTRL+P	Print
CTRL+Q	Cancel (Quit) without saving. If changes are detected you will be asked if you want to save them.
CTRL+R	Replace
CTRL+S	Save
CTRL+SHIFT+E	Reverse Incremental Search
CTRL+SHIFT+C	Classes
CTRL+SHIFT+F4	Close Solution
CTRL+SHIFT+G	Insert new GUID
CTRL+SHIFT+L	Toggle All Folds

CTRL+SHIFT+M	Toggle Fold
CTRL+SHIFT+N	New Solution
CTRL+SHIFT+O	Open File with Redirection
CTRL+SHIFT+P	Open Solution, Project or Application
CTRL+SHIFT+S	Save All
CTRL+SHIFT+U	Reload File
CTRL+TAB	Open Active Window Pop-up
CTRL+V	Paste
CTRL+X	Cut
CTRL+Y	Redo
CTRL+Z	Undo
F1	Contents
F2	Next Bookmark
F3	Find Next
F4	Properties
F5	Start Debugging
F6	Continue Debugging
F7	Toggle breakpoint
F8	Build Solution
F9	Tab Order Assistant
F10	Step over
F11	Step into
F12	Data / Tables
Mouse-Double Click	On Title of any window – toggles dock/undock
SHIFT+ESC	Toggle IDE Layouts

SHIFT+F11	Step out

Icon Quick Reference Guide

The new Clarion 7 IDE is packed with new visual indicators that will help you visually identify key features and characteristics of many parts of the development environment.

Here is a summary list:

Dictionary Explorer

 Standard Table

 Table with triggers

 ALIAS table

 Column (Field)

 Column GROUP

 Primary Key

 Standard (non-Primary) Key

 Ascending Key Component

 Descending Key Component

 One to Many Relation

 Many to One Relation

Code Completion

Code Completion is a new feature in the Clarion 7 Text Editor. You can use this feature while entering a language statement or a part of a namespace, and a drop list appears with a set of valid choices.

Icons are used to help identify different types of language constructs:

Class

Delegate

Enum

Interface

Method

Struct

Intermediate name – This icon implies that this item requires additional items or parameters after selecting it.

General Tips Using the New IDE

Current Directory

Something that can be a little confusing with the new IDE is the idea of a current directory.

In Clarion 6 and prior versions, you could select the current directory, and every time you opened a file, the file dialog would start in that directory.

In Clarion 7, there is no longer the concept of a current directory. This is necessary because you now have solutions that contain multiple applications each of which can be in a different directory.

So the question then is, when I open a file in which directory will the file dialog start in?

The answer depends on what is active in the IDE.

If you are opening a project/solution or application, then the file dialog always starts in your default project location as specified in **Tools/Options/General/Projects and Solutions** dialog.

If you open a file via **File/Open/File** or **CTRL+O** then the initial directory will be the directory that the currently active tab's file is found. If there are no open tabs, then the directory will be the directory of the currently active project/application. If no solution is loaded, then the initial directory will be the directory where the IDE was started from.

Designer Related Tips

Custom Color dialog
For each property that requires a color attribute, there is a drop list that displays a variety of default colors to select from. However, if you need to choose a custom color, select the Custom tab, and RIGHT-CLICK on any color element in the bottom two rows of this dialog.

Multi-DLL projects in Clarion 7

The following topic documents the sequence of recommended steps needed to link a multi-DLL source code solution in Clarion 7.

Overview

The Clarion 7 project system has been updated and upgraded to use the MSBuild project engine. MSBuild is the new extensible, XML-based build engine that ships with the .NET Framework 2.0

All projects compiled in Clarion 7 are stored in solution files (*.SLN). These files can store multiple projects ,and allow multiple resources to be referenced and included.

Think of the projects used in earlier versions of Clarion as a subset of any solution.

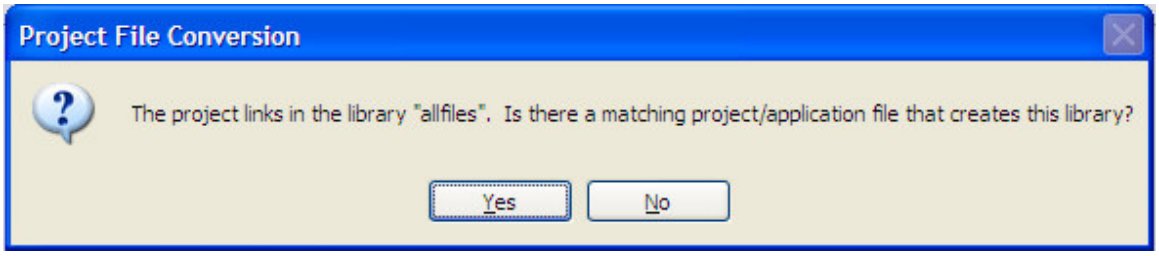
How to create a Clarion 7 Multi-DLL Project

To create a multi-DLL project, you will need to create a solution which references all the DLLs and Executables that are needed. Here is a step-by-step quick-start guide to this process:

Note: This sequence applies to a hand coded multi-DLL project, or projects that have been exported from an application. The sequence of creating a multi-DLL solution from existing applications will vary slightly.

1. Locate the main project file (the .PRJ file) of the multi-DLL project, and open it in the Clarion 7 IDE. This should be the project file that creates the executable (EXE). To do this, select **Open > Solution, Project or Application** from the IDE **File** menu.

The **Project File Conversion** dialog will be displayed at this time**.** For example:



Press **Yes** to find a matching project (PRJ) for all of the library files that are detected. This will complete the multi-DLL project import, and you can now skip down to step ***8***.

If you are not sure if a corresponding project exists for a library, or wish to manually import your project references, press **No** in the **Project File Conversion** dialog, and proceed to step ***2***.

2. In the **Projects** window, right click on the project node, and select **Add > Referenced Projects...** .

3. In the selection dialog, click on the **cwproj Browser** tab.

4. Click on the **Browse** button.

5. In the **Select project(s) to reference** file dialog used to select a project change the **file filter** to *Legacy Clarion Project Files.*

6. At this time you can add all DLL projects needed by performing a multi-select in the **Select project(s) to reference** dialog.

7. Click **OK** to go back to the **Add Reference** dialog, and click **OK** again to close this dialog window.

This will convert all the projects to Clarion 7 *cwproj* files, add the projects as references, and add them to the new solution. The MSBuild engine will automatically resolve priorities and references from this point.

8. Build the solution (**Build > Build Solution**). If you have already built the solution before, you may want to "clean" the solution first (Build > Clean Solution).

Clarion 7 creates three sub-folders with every solution, located in the project folder. They are BIN, MAP and OBJ. In each of these folders, you will see Debug or Release (or both), depending on the configuration mode that you have selected. Cleaning a solution removes all files from these sub-folders.

Development Environment (IDE)

IDE Document Conventions

Many areas of the IDE are divided into tabbed document areas. Example of this are the Start Page, Text Editor, and Dictionary Editor Quick View.

If a document tab is marked with an asterisk (*), it indicates that the document or section has changed and needs to be saved.

If a document tab is marked with a plus sign (+), it indicates that the document or section is currently in a read-only state, and cannot be modified. This can happen in a dictionary file if the user has read-only rights.

General Environment Menu Commands

The General Environment Menu refers to the menu commands of Clarion's IDE (Integrated Development Environment) when no special area is active (Dictionary Editor, Application Generator, etc.).

File Menu

New	Opens the **New** dialog, which lets you create a new dictionary, application, solution, project, or other file type. Select from a **New >File**, which calls the New File dialog, or **New > Solution, Project or Application**,which calls the New Project dialog.
Open File	Allows you to open a file (Dictionary, Appplication, Source, etc.). If the file is a dictionary (*.DCT) it will open the Dictionary Editor, if the file is a source file (*.CLW, *.CLN) it will open the Text Editor, and if the file is an application (*.APP) it will open the Application Editor.
Open Project or Application	Use this option for application files. This will allow the application to be compiled if needed. This option looks for all known project formats that you have installed, and will load them appropriately.
Open File using Redirection File	You can also open any file using the current active IDE redirection file. (**Open > Open File using Redirection File**). Enter the name of the target file, and all redirection paths will be searched in an attempt to locate the file.
Close	Closes the active file or solution.
Reload File	Allows you to reload an active project file. This may be useful if you have modified the target file outside of the IDE, and need to refresh it.
Save	Saves the active file.
Save As	Saves the active file under a new name which you specify.
Save All	Saves all the active files.
Print	Prints the active document.
Print Preview	Calls the Print Preview dialog. From here, you can review a file's contents before printing.

Recent Files	Displays a list of recent files accessed by the IDE.
Recent Solutions	Displays a filtered list of recent solutions accessed by the IDE.
Exit	Quits the program.

Edit Menu

Undo	Reverses the most recent editing action.
Redo	Reapplies the most recent editing action.

Cut	Deletes the selected text from the document and places it in the clipboard.
Copy	Places a copy of the selected text from the document into the clipboard.
Paste	Pastes text from the clipboard into the active document, at the insertion point.
Delete	Deletes the selected text from the document and does not place it in the clipboard.
Insert Color	Adds a color equate to the active document.
Insert new GUID	Adds a new Globally Unique Identifier to the active document

Format:

Remove Leading Ws	Remove all leading white spaces up to and including a new line
Remove Trailing Ws	Remove all trailing white spaces up to and including a new line
To Upper Case	Converts selected text to upper case
To Lower Case	Converts selected text to lower case
Capitalize	Converts selected text to upper case *only* the first letter of each word selected.
Sort Lines	Opens a dialog that allows you to resort all selected lines in alphabetical acending or descending order. You also have the option to **Remove Duplicate Lines**, make the sort **Case Sensitive**, and **Ignore any trailing whitespaces** encountered.

Tabs to Spaces	Converts any tab characters in the selection to spaces.
Spaces To Tabs	Converts any space characters in the selection to tabs.
Leading Tabs To Spaces	Converts any leading tab characters in the selection to spaces.
Leading Spaces to Tabs	Converts any leading space characters in the selection to tabs.
Comment region	Comments all lines from a selected region.
Uncomment region	Removes comments from all lines in a selected region
Indent	Indents the selected text based on the editor's setup options.

Folding:	*Folding* is a mechanism that allows regions of text to be hidden, often replaced by a summary line of some sort.
Toggle Fold	Toggle selected fold between the full text or text summary.
Toggle All Folds	Toggle all folds between the full text or text summary.
Show definitions only	Select this option to only fold all classes and prototypes. Select the **Toggle All Folds** option to cancel this action.

Select All	Selects all text in the active document.
Enable Code Completion	Toggles the use of Code Completion in the active source document. See the Code Completion topic for more information.
Word Count	Displays a word count of the active document.
Structure Designer	Opens the Structure Designer tool, allowing you to create both WINDOW and REPORT structures.

View Menu

The View Menu can be used to display a wide variety of viewers (or "scouts" as #Develop likes to call them). These viewers provide a variety of information that is pertinent to your current project or solution. Select from one of the following options.

Solution Explorer	Opens the Solution Explorer window, which displays all pertinent files related to the target solution. This includes source, drivers, and other resources.
Classes	Displays all class structures references by the current active project or solution.
Toolbox	Displays a variety of read only lists, which can be used to drag and drop values into any valid target (i.e., source files)
Errors	Displays an error list that contains any errors or warnings generated from the last active build
Task List	Displays a list of special comments inserted into source that define a list of tasks specified by the developer. See Tools > Options for additional information.
Output	Displays the output of the last built project or solution. This window is displayed automatically when any project or solution is built.
Properties	Displays a list of properties associated with the active source file. Sidebar help is displayed below based on a selected property.
Data Sources	Displays a hierarchal tree of files, queues, groups and other structures detected in the current active project. This pad is used to quickly populate components in windows and reports.
Document Outline	Currently not used in the Clarion IDE
Control Templates	Provides a list of all control templates in the active template set. When an application window or report is active, you can drag a control template in the list and populate it on the window.
Data / Tables	Displays a list of all active data components in the active dictionary and application.
Applications	Provides a view of all opened applications in the IDE
Tab Order Assistant	Displays all controls in an active window or report. Its purpose is to allow easy changes to any control's **TabIndex**, **Name** and **Text** properties.
Tools	The Tools menu item displays a submenu of Viewers or other windows (Panes) that were added to the basic IDE.

Debugger Windows	Refers to the Clarion.NET Debugger. You have a variety of Views to choose from. (Not applicable to Clarion 7)
Data Dictionary	Accesses all Dictionary related Views and Pads. Currently, the only applicable pad is the Dictionary Changes Pad, which displays conversion results from earlier

Clarion versions.

Show start page Opens the IDE Start Page, which lists recent solutions, a change log of the current build, an authors credit list, and a call for contributions from developers. Also the first button to the left of the toolbar.

Fullscreen Switches to a full screen mode, removing all toolbars.

Project Menu

The Project Menu offers a variety of options that allows you to build, run, modify and configure the current active project. This menu is only visible when a project is selected.

Although the IDE allows for different target outputs (C#, Boo, VB, etc.) Clarion 7 is optimized for the Clarion language. Select from the following options:

Add > New Item	Add a new MEMBER or PROGRAM file to the active project.
Add > Existing Item	Add an existing file to the project. This can be any valid resource that is detected or needed by the project.
Add > File Driver	Add a file driver library to the project.. The File Driver dialog is displayed.
Add > Library, Object, Resource, Manifest	Add an existing component to the project. This can be any valid resource that is detected or needed by the project.
Add > Referenced Projects	Add an additional project reference to your existing project. The IDE looks for valid project file extensions.

Run Project	Runs the active project, if applicable. Target must be an executable.
Debug Project	Runs the active project in Debug mode, if applicable. Target must be an executable. Invokes the Clarion 7 Debugger.
Set as StartUp Project	Sets the project as the one the IDE will execute when you click the Run button. Your solution may have more then one executable target project.

Project Options	Opens the *Project Options* window.

Build Menu

The Build Menu acts on the current project or solution and creates the output designated from the project settings. Select from the following:

Generate and Make Current Application	Generate source, compiles and builds the current application.
Build Solution	Builds the active solution. The **Build Solution** option compiles only those project files and components that have changed since the last build.

Rebuild Solution	The **Rebuild Solution** option builds all project files and components irrespective of the changes made to them.

Clean Solution	Deletes all files from the target project folders. This action uses an XML based FileList that was generated by the first Build activity. If you attempt to clean a solution that has never been built, you will receive a project warning that it could not be cleaned.

Set Configuration	Choose **Debug** or **Release** Mode. See the Project Options window for more information.
Set Clarion Version	Specify which version will be used for compiling and linking the active project. The (default) is based on the current redirection path, or specify a specific version to use. See *Multiple Version Support* for more information.
Edit Configurations/Platforms	Calls the Configuration Editor. This feature is not implemented yet.

Debug Menu

Make and Run All	Generates source, compiles and build the active solution, and runs without the Clarion 7 Debugger.
Make and Debug All	Generates source, compiles and build the active solution, and runs with the Clarion 7 Debugger.
Start Debugging	Runs the active solution in the Clarion 7 Debugger.
Start without debugging	Runs the active solution from the IDE without the Clarion 7 Debugger.
Attach to Process	Not applicable with Clarion 7. Selecting this item will produce no results. The IDE, being based on .NET, can always attach to a .NET process, even in Clarion 7 without the Clarion.NET/Clarion# support., so currently this menu item cannot be removed or disabled.
Stop Process	Not applicable with Clarion 7.
Detach	Not applicable with Clarion 7.

Toggle Breakpoint	Not applicable with Clarion 7.

Search Menu

Find	Opens the **Find** dialog in the *Search and Replace* window
Find Next	Opens the *Search and Replace* window where the active search continues.
Find Next Selected	Searches a target string in the selected text of the active document.
Replace	Opens the **Replace** dialog in the *Search and Replace* window
Incremental Search	You can search an individual document or window incrementally by entering a search string character by character, then observing the matches found as the search string lengthens. First you enter incremental search mode, then type in the search text. As you type in each character the document is searched from the current cursor position and the first match it finds is highlighted.
Reverse Incremental Search	The reverse incremental search will search from the current cursor position to the start of the document.

Toggle Bookmark	Toggles bookmark on selected line of active document
Prev Bookmark	Selects previous bookmark in active document.
Next Bookmark	Selects next bookmark in active document.
Clear All Bookmarks	Clears all bookmarks in active document.

Go To Line Number	Opens the *Go to* dialog, which allows you to jump to a line number, class name, or file name in the active document.

Go To Matching Brace	Jumps to the matching brace of highlighted starting brace in the current active document.

Tools Menu

<custom utilities>	Displays a variable list of custom programs that are added to the External Tools section. See **Tools > Options> Tools**

AddIn Manager	Opens the AddIn Manager, which allows you to add additional IDE add-in tools.

AddIn Scout	Opens the AddIn Scout, which provides additional detailed viewing options for all add-ins installed.
Component Inspector	Explore and search classes and types from any .NET assembly or COM component.
Regular Expressions ToolKit	Opens the Regular Expressions ToolKit (not applicable to Clarion 7)

Edit Template Registry	Opens the Application Generator's Template Registry. Register and Maintain your Registry here.
Edit Template Registry using alternate RED file	You can now edit the template registry when you have a local redirection (.RED) file that redirects the location of the template registry (.TRF) file to a different location than the standard .RED file.

Browse Table	Opens the Database Browser utility, that allows you to view and edit any IDE registered table
Edit Base Redirection File	Loads the Redirection File in a document window, ready for editing.C7 and C6 Red Files
Register File Drivers	Calls the Database Driver Registry, which lets you register database drivers.
Register Database Systems	Opens the Registered/Unregister Database Engines dialog.
Application Options	Open's the Options window for the Application Generator.

Auto code generation	Opens the Auto code generation tool.
Quick XML doc	Opens the Quick XML documentation tool (not applicable in Clarion 7). See also: XML Documentation
Subversion – Export	Used with the Tortoise Subversion control client
Subversion – Checkout	Used with the Tortoise Subversion control client

Options	Calls the Options Dialog, which lets you customize the appearance and behavior of the active tools in the IDE.
Connect to Device	For Enterprise Edition users, this option allows you to test your connection to installed mobile devices and emulators.

Window Menu

Next Window	Arranges open document windows side by side in a vertical orientation.
Previous Window	Arranges open document windows side by side in a horizontal orientation.
Split	Arranges open document windows in overlapped fashion so that all caption bars are visible
Close All Documents	Arranges iconized windows along the bottom of the Clarion Application frame.
(Window List)	Lists all open document windows by their caption bar text according to the order they were opened. Choosing a window from the list brings the window to the top.

Help Menu

Content	Opens the Clarion HTMLHelp and displays a list of main topics.
Context Help	Opens the Clarion HTMLHelp and displays a list of main topics.
.NET API	Displays a sub menu of topics shown below, intended to call the .NET Framework help system.
Dynamic Help	This window provides links to topics that are specific to the current area you are using or task you are trying to complete within the IDE. The IDE tracks the selections you make, the placement of the cursor, and the items with focus as you interact with what's on the screen. The Dynamic Help window then filters through the topics available in all collections that have been plugged into Clarion 7 Help Collection (see the Tools Options window) to display relevant information. So as you type or focus on different windows, the Dynamic Window updates the information available to you.
Contents	Opens the current MSHelp 2 active collection and displays a list of main topics.

Index	Opens the Index window of the MSHelp2 active collection.

Search	Opens the **Search** dialog in the MSHelp2 application, allowing you to search for help topics containing a specific keyword.
Index Results	Displays a history of the Index entries recently accessed.
Web	Opens a submenu with many useful links related to Clarion and the .NET Framework.
About	Displays the program name, version, registration, and copyright information.

Toolbar Menu

Start - Opens the Start Page, which allows you to pick from a list of projects, applications, dictionaries, etc.

New Solution - Opens a new solution. The **New Project** dialog is opened.

Open Solution - Opens an existing solution.

Close Solution - Closes the active solution.

Add Existing Project - Add an additional project to the active solution.

New File - Opens any new file. The New File dialog is opened. This can be a new source file, text file, just about any file you want.

Open File - Opens any IDE related file. Select from the File Type drop list for the ist of available file types.

Save - Saves the active file to disk.

Save All - Saves all active files to disk..

Cut deletes the selected text from the document and places it in the clipboard. **Copy** places a copy of the selected text from the document into the clipboard. **Paste** pastes text from the clipboard into the active document (at the insertion point).

Delete - Deletes the selected text from the document and places it in the clipboard.

Undo - Reverses the most recent editing action.

Redo - Reapplies the most recent editing action.

Generate Selected – Active when any application is opened in the Application Editor. Generates source for the application that currently has focus.

Generate All Opened – Generates source for ALL applications opened in the Application Editor, or marked as opened in the Applications Pad.

Generate All - Generates source for ALL applications opened in the current active Solution. See the Applications Pad topic for more information.

Generate and Make - Generate source, compiles and builds the current application.

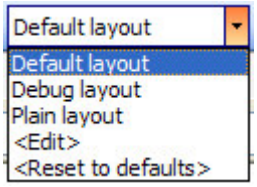
Build Project - Build active project or buffer (when no project is open) (does autosave)

Rebuild Open Project - Rebuilds the current project. This action cleans the project folders before beginning the build.

Run with Debugger - Runs program in the debugger. This option will generate source for applications opened for editing, and then compile, and build all prior to the Run if needed.

Run without Debugger - Runs program from IDE. This option will generate source for applications opened for editing, compile, and build all prior to the Run if needed.

Run the Start Up Project – Runs the start up project without any other action (generate or build).



Select Layout - Selects what views and toolbars are opened.

Navigate Back - Allows you to navigate back to a specific document and search result area.

Navigate Forward - Allows you to move ahead to a specific document and search result area. Becomes active as soon as you navigate back.

Enable Code Completion - Activate the Code Completion feature of the Text Editor.

Comment Region - Adds comments to a selected region.

There is a sub menu of options available here. You can specify when a group of selected lines are commented:

Add a Debug ? comment mark toi each line

Adds an optional OMIT or COMPILE structure around the selected lines.

Uncomment Region - Removes comments from a selected region.

An additional option is also available to remove Debug comments from the selected region.

Open Designer - Opens the Window/Report Designer

Toggle Bookmark - Toggles bookmark on selected line of active document

Previous Bookmark - Selects previous bookmark in active document.

Next Bookmark - Selects next bookmark in active document.

Clear Bookmarks - Deletes all bookmarks in active document.

General Setup and Navigation

Core IDE Elements

Clarion's development environment is a program that helps you write other programs. The development environment manages all the files (source code, executables, resources, etc.) and all the processes (editing, compiling, linking, etc.) needed to successfully develop Windows programs. The *Getting Started* and *Learning Clarion* books provide a wonderful general introduction to Clarion's development environment.

The tab controls on the Pick List, and New and Open menu options, redirect you to the core components of the Clarion IDE (Integrated Development Environment):

Application (*.app)
Selecting the Application tab control or menu item, and opening a file with a .APP extension, starts the Application Generator. By convention, a Clarion application refers to a project whose source code is generated by templates and whose instructions for compiling and linking are stored within the application file (.APP), rather than in a separate Project System file (.PRJ).

Dictionary
Selecting the Dictionary tab control or menu item, and opening a file with a .DCT extension, starts the Dictionary Editor. Data Dictionaries are created and maintained here. The Data Dictionary is the central repository for information about your application's tables, columns, keys, and relationships. The information stored includes how and where the data is stored on disk, as well as how the data is presented to end users on reports and computer screen. This will usually be the first place to visit when starting a new project.

Project
By convention, a Clarion project includes some set of source code, plus the instructions (.PRJ file) for compiling and linking to produce an executable program. Project typically refers to a manually coded program rather than a template generated program. The Clarion Project System visually manages the project information. It maintains tree diagrams of the source files, external libraries, resources, and other project components.

Database
Clarion support a wide variety of database formats. This tab control provides access to the Database Browser, which in turn provides you to direct access to tables without the need of creating an application. The Database Browser is designed to allow developers free access to their tables.

As you open a particular table, you will need to select the particular database type (i.e., TopSpeed, Pervasive, MS-SQL, ODBC, etc.). The information provided to you in the Database Browser varies with different database drivers.

Source
This tab control accesses the Text Editor. If you allow the Application Generator to write most of your source code, you will probably only use the Text Editor to write your embedded source code (see Embedded Source Dialog). If you write your own source code "from scratch", you will probably use the Text Editor extensively to create and manage your code. In addition, the Project System is integrated into the Text Editor, allowing you to effortlessly generate executables, libraries, and dynamic linked libraries (DLL).

Report
Selecting the Report tab control, and selecting or opening a file with a .TXR extension, starts the Clarion Report Writer. The Clarion Report Writer lets you create sophisticated reports from your data files, without the need of creating an application. See the Report Writer documentation for more information.

All
Selecting this tab provides you with a history of all files that you have accessed in the Clarion IDE. Selecting a file results in a "smart" open, redirecting you to the correct area of the environment.

New File

The **New File** dialog is designed to create and then edit a wide variety of file types useful in Clarion development and more.

Use the buttons in the upper right to switch from a small or large icon display.

Some file types enable an additional properties dialog with other file options.

Highlight the target file, and press the Create button to create the new file type.

Note:

If you need to create a new Clarion or Clarion.NET project or application, use the **New >Solution, Project or Application** menu option.

Register File Drivers Dialog

The Clarion database drivers are pre-registered for you. Should you remove one and need to reinstall it, or if you obtain a new driver, you must use this dialog box to register the new driver. To do so:

1. Choose **Tools** ▸ **Register File Drivers**.

2. Press the **Add** button in the **Register File Drivers** dialog box.

3. Select the file driver, which normally has a DLL file extension, from the **Add Database Driver Open File** dialog, and press **OK**.

The database drivers belong in the ..\BIN subdirectory.

4. Press **OK** to close the **Register File Drivers** dialog.

The dialog contains these additional buttons:

Remove

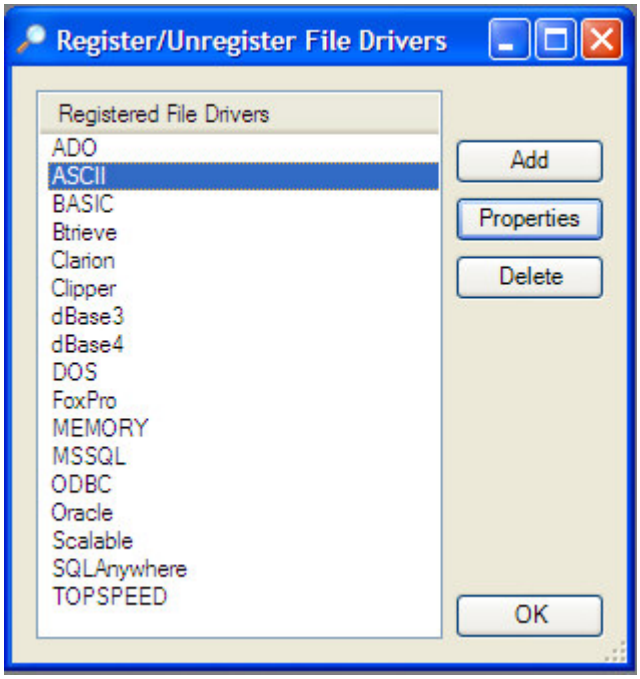
Deletes the selected driver from the registry. This does not delete the .DLL file from your drive.

Update

Updates the registry information for the selected driver.

Reminder: when distributing your application, you must ship the database driver library files used by your application. The file names are listed within this dialog.

At the time of this release, here are the available drivers and associated DLLs that needs to be registered:



Register Database Systems Dialog

This window displays the database servers that are pre-registered for you to use with the Dictionary Synchronizer. Should you remove one and need to reinstall it, or if you obtain a new driver, you must use this dialog box to register the new driver. To do so:

1. From the main environment menu, choose **Tools** ▸**Register Database Systems**.
2. Press the **Add** button in the **Register/Unregister Database Engines** dialog box.
3. Select the appropriate library, which normally has a DLL file extension, from the **Select Database Engines** dialog, and press **OK**.

The database drivers belong in the *..\BIN* subdirectory.

4. Press **OK** to close the Dictionary Synchronizer Registry dialog.

The dialog contains these additional buttons:

Delete
Deletes the selected driver from the registry. This does not delete the .DLL file from your drive.

Start Page

The **Start Page** is the default work area, or main page of the Clarion 7 IDE. Developers using earlier versions of Clarion may recognize this as the new "Pick List". This window is always displayed by default, unless you have activated the option of loading a default project.

The Start Page dialog lists all of the most recently used files in a list box categorized (filtered) by **Solutions (Projects and Applications)**, **Dictionaries**, **Diagrams**, **Tables**, and **Source Files**. See Core IDE Elements for an expanded explanation of each of these categories.

Each of these categories displays a pick list of the most recently used files of that type:

The **Start Page** also provides the following options for each section:

Open	Lets you open a file not on the Start Page list.
New	Lets you create a new file.
Browse	Found in the **Tables** category, and used to select any database table
✕	Closes the Pick List without selecting a file or IDE component.
F1 Key	Calls this window.
Remove missing recents from the list	Check this box to automatically remove from the list all items where a solution file is not found. This is useful if you have renamed or removed certain solutions prior to opening the Start Page.

The Start Page also displays a toolbar that is similar to standard Internet Browser functions. To access any web site, enter the URL in the address bar.

Use the arrow icons to navigate forward and back between web pages. You can also stop or refresh any web page activity, navigate directly to the SoftVelocity home page, or access a web search page.

To the right of the address bar, the New Window icon is used to open a new browser window.

Template Registry Dialog

This window is opened from the IDE Tools Menu (**Tools > Edit Template Registry**)

Template files (*.TPL) drive the Application Generator. Each procedure template contains generic or "model" code. The templates are interactive--they process the information you specify when you design the application within the IDE. Clarion evaluates the template file twice:

1. Before creating your application, Clarion pre-processes the template file and stores the information in the *TemplateRegistry.TRF* file. Pre-processing occurs only when the Application Generator detects a new or changed template.

When it pre-processes the template file, the Application Generator stores a list of all the information you must provide each procedure. It also determines the points at which you can embed your own Clarion source code to customize a procedure. The registry file contains the default windows, dialogs, menus, report designs, default data, and formulas. In the design process, you customize these defaults.

2. At code generation time, the Application Generator evaluates the information you provide in the design process--from the data dictionary, and the .APP file--then processes it along with the template language statements and symbols in the *TemplateRegistry.TRF* file to generate your source code.

Each template can contain multiple types of procedure templates from which you select to create the procedures in your application. Before you can use a template it must be in the Template Registry.

You can see which template registry file you are editing by hovering the mouse over the **Template Registry** tab

The **Template Registry** dialog provides toolbar buttons for file maintenance options for the registry:

The Register, Unregister, Enable, and Disable buttons are not available in Multi-Developer mode. To change this setting, select **Tools** ▸**Application Options**.

Save and Close 

Saves the current registry changes and closes the registry

Cancel 

Closes the Template Registry dialog window without saving the current changes.

Save 

Saves the current registry changes without closing.

Register

Calls the Open File dialog, which allows you to register a template (.TPL) file.

If you encounter any error during the registration process, use the IDE Outpad Pad to view and correct any template errors.

Unregister

This button deletes the currently highlighted template class from TemplateRegistry.TRF.

Edit Definition	This button displays the Template code (the .TPL or .TPW) in the Template Editor. You may edit the code in this window. If the currently highlighted item in the Template Registry tree is a module, the text window opens to the first line of the MODULE definition. If a procedure, it opens to the first line of the PROCEDURE. If the highlighted template is one of several in a single file, it opens to the first line of the highlighted template.
Enable	This button enables the currently highlighted template class or procedure (if you had previously disabled it).
Disable	This button disables the currently highlighted template class or procedure, which makes it unavailable to your application.
Expand All	This button expands all template classes to reveal all template sections for each template class registered.
Contract All	This button contracts all template classes to Class definitions only. Useful when there are a large number of template classes registered.
Help (F1 Key)	Displays this window.

Prior versions of Clarion also had a **Properties** button that allowed access to the properties of any procedure template. To access these properties in this version, simply DOUBLE-CLICK (or press the ENTER key) on the target highlighted procedure template.

Tools Options Dialog

To personalize your editing environment, customize the appearance, cursor behavior and other items use the **Tools Options** dialog.

To view the dialog, choose **Tools** ▸ **Options**. From the main IDE menu. Select the *Text Editor* tree item, which reveals the following set of sub-options.

General

Markers and Rulers

Behavior

Clarion Specific Options

Code Completion

XML Options

XML Schema

Highlighting

Tools Options Dialog

The Options dialog, opened in the Clarion IDE Tools Menu (Tools ▸ Options…), contains a wide variety of options that are designed to setup the working IDE.

The following options are available:

General:

UI Language

Appearance

Fullscreen

Load/Save

Task list

Output Window

Projects and Solutions

File Format Associations

Coding:

Code Generation

Edit Standard Headers

Code Snippets

Component Inspector:

General

Object Tree

Type Handlers

Text Editor (See the section Text Editor section in this document)

Tools:

External Tools

Code Analysis

Code Coverage

Help 2.0 Environment

Windows Forms Designer:

General

Grid Options

WINDOW Structure Designer:

General

Grid Options

REPORT Structure Designer:

General

Grid Options

Device Tools:

General

Devices

FormFactors

Web Forms Designer:

General

Clarion:

Clarion for Windows:

General

Versions

Dictionary Editor Options

Dictionary Diagram

Application General Options

Tools Options - UI Language

Select a base language to work with in the IDE. All core menu items and dialog prompts controlled by the IDE will be affected.

Tools Options - Appearance

The **Appearance** dialog is used to control the overall appearance of the IDE. Choose from the following options:

Show file extensions in the project scout
Clarion 7 has its own project interface. This option is not applicable in Clarion 7.

Select your preferred ambience
The Ambience type you select will be used in the quick class browser (the combo boxes located above the text editor) and for the tooltips and code-completion. It controls the way the tooltips are shown.

For example, for a given method, the C# Ambience returns "string ToString()", the VB Ambience "Function ToString() As String", etc.

Select the Clarion or Clarion# Ambience to display the tooltips in the syntax of the target language.

Prefer project's ambience if possible
Check this option to allow the IDE to switch an ambience based on the type of project you currently have active.

Show status bar
Check this option to show the standard status bar in the IDE.

Show tool bar
Check this option to show the IDE tool bar. If not active, all of the options you need are contained in the IDE menu.

Use blue-style (Office 2003-like) for menus and toolbars
Check this option to display the "blue style" menus. Leaving this option off shows a standard "grey" appearance.

Tools Options - Fullscreen

Fullscreen mode is implemented through the IDE **View** Menu. The options below control what you wish to display while in that mode:

Hide main menu
If you activate this option, you can also control its display when the mouse cursor moves over the area.

Hide toolbars

Click this option to hide the IDE toolbar in Fullscreen mode

Hide tabs
Hide vertical scrollbar of text editor
Hide horizontal scrollbar of text editor

These options above are currently not implemented (permanently disabled)

Hide status bar
This option hides the IDE status bar in Fullscreen mode

Hide Windows Taskbar
This option is currently not implemented

Tools Options - Load/Save

The **Load/Save** options control how all documents are handled in the IDE. Documents are listed in the **Start Page** *Source Files* link, and in the **File > Recent Files** IDE menu item.

As you create a new document, the path is specified as you save the document for the first time.

Load user-specific settings with the document
When this option is active, your user-defined settings, stored in the Documents and Settings Application Data section, will be applied. If this setting is off, the IDE's internal defaults are used.

Detect external changes to files
This option displays a dialog to reload a document when changes have been detected from an external source (i.e., an external editor).

Auto-load changes if saved
Specify to reload a document if changes in the document have been detected.

Always create backup copy

Check this option to automatically create a backup copy of all documents.

Line terminator style
Select the line terminator style of documents that you save. The *Windows* line terminator character is CR + LF, *MacIntosh* style uses just CR, and *Unix* style uses a plain LF only.

Maximum number of items to display in Recent File
Use the spin box to set how many files to list in the **Recent Files** menu section and in the **Start Page** *Source Files* link.

Tools Options - Task list

The options in this dialog control the keywords that the Task List tool will search for. For example, if you have the "FIXME" string embedded in your document, the Task List tool will locate the line number when executed.

In summary, you can set custom comments in any Clarion project, and use the task list to recall their locations for future editing.

Token List
Contains the current list of keywords used in the Task List

Name
Allows you to edit existing tokens, or add and edit new ones.

Tools Options - Output Window

The options here control the appearance of the output window. This is the window that displays Build results and Errors (if any).

Word Wrap
Check this option to allow word wrapping of the information displayed in this window.

Font
Set the font used in the output window here.

Tools Options - Projects and Solutions

The options in this dialog control the default location and error options of new Clarion 7 and Clarion# projects and solutions created.

Settings

Use Directory of Last Selected Directory as Default Project Location
When checked, the last directory accessed by the IDE will override the **Default project location.**

Default project location
Specify the default directory for all new projects created.

Use solution folder as default when adding a new project to the solution
Check to maintain the current solution folder as you add new projects.

Load previous solution on startup
Check to auto-load the last project/solution that you were working on in the IDE as you restart the IDE in a new session.

Close Start Page when loading a solution
Check to automatically close the Start Page anytime a solution is loaded.

Build and Run options

The Clarion IDE supports Builds on a separate thread or a separate process. By default, the IDE Builds on a separate process. On multi-CPU or multi-core CPU machines this will normally improve the performance of the system as the Build action will normally execute on a different CPU that is different to the one the IDE is using. However, due to the added overhead of interprocess communication, it may be that on some machines it is better to run the Build as a separate thread within the IDE. To disable the Build activity in a separate process, uncheck the **Run build as a separate process(es) rather than as a thread** check box.

Check the **Show error list pad if build finishes with errors** to show the Error list by default on a Build error. If unchecked, the Output window will display instead with a more verbose error listing.

Output Details
Select the type of messaging output that you want during the Build process. The following options are currently available:

Quiet	Only display "Build started.", "Build finished successfully." messages, plus errors and warnings.
Minimal	Same as Quiet
Normal	Same as Quiet, plus messages indicating which source files are being compiled.
Detailed	Same as Normal, plus details about which files are included when compiling and which files are linked when linking
Diagnostic	Same as Detailed

Tools Options - File Format Associations

This dialog contains a comprehensive list of all file types registered in the IDE. Several file types are enabled (checked) and associated with Clarion by default.

File Associations simply trigger the Windows O/S to recognize certain file formats and associate them with the Clarion IDE. Uncheck certain files if you want to associate them with another program.

Coding - Code Generation

This option should not be confused with the built-in code generation options of Clarion 7 and Clarion#

The Code Generation options here refer to the base SharpDevelop support for C#.

Given an interface that you want to implement, it can be a bit tedious to write all the class methods just to get your code to compile, especially if the interface has a large number of methods. SharpDevelop can save you time by generating code for you.

The following link has some additional help regarding this feature:

http://community.sharpdevelop.net/blogs/mattward/archive/2006/07/05/FeatureTourCodeGeneration.aspx

Tools Options: Coding - Code Snippets

Code snippets in this IDE area are very similar to standard editor macros. The "snippets" defined here can easily be applied in any file matching the group extension.

Code snippets are applied using the following syntax:

code snippet + <SPACE>

Code snippet is the name defined in the *Snippet* entry.

In the Text Editor, press **CTRL + J** to pop up a list of defined code snippets. You can select any item in the pop up list, and press the <ENTER> key to apply it.

Code snippets defined are case sensitive. In addition, the file extensions defined in the group are also case sensitive. For example, add **.CLW;.Clw;.clw** in order to be sure that you could pop up a Code Snippets regardless of the case of the Extension on the file that you are editing.

Tools Options: Coding - Edit Standard Headers

Standard Headers are textual information that loads by default when new files are created.

Standard headers for C#, VBNET, C++.NET and of course, Clarion, are listed here.

This is the default header for new Clarion (CLW) files:

```
OMIT('***')
* Created by Clarion 7.0.
* User: ${USER}
* Date: ${DATE}
* Time: ${TIME}
*
* To change this template use Tools | Options | Coding | Edit Standard Headers.
***
```

At press time, here is the complete list of "User defined" IDE elements that you can use in the header:

"USER", "DATE", "TIME", "ITEMPATH", "ITEMDIR", "ITEMFILENAME", "ITEMEXT", "CURLINE", "CURCOL", "CURTEXT", "TARGETPATH", "TARGETDIR", "TARGETNAME", "TARGETEXT","PROJECTDIR", "PROJECTFILENAME", "COMBINEDIR", "COMBINEFILENAME", "STARTUPPATH"

Each element should be prepended with a $ character, and surrounded by curly braces "{ }".

Component Inspector

The options in this section affect the Component Inspector tool, which is found in the IDE Tools menu.

This tool is used to explore and search classes and types from any .NET assembly or COM component. You can also:

Trace events on any object.
Create any object from an assembly.
Create controls on the design surface, move, resize or embed them in other Controls;
Search for types or object content.
Examine or alter any field or property in the created objects.
Execute any method on the created objects
Execute an application, class, or control without writing any code.

More information can be found at the following link:

http://community.sharpdevelop.net/blogs/mattward/archive/2006/06/20/UsingTheComponentInspector.aspx

and also:

http://community.sharpdevelop.net/blogs/mattward/archive/2006/05/21/ComponentInspector.aspx

Component Inspector - Object Tree

The Object Tree options affect how data is displayed in the Component Inspector's Object tab, allowing you to show or hide properties, methods, events for the objects on display.

Show Members

In the object tree, check the members that you wish to display in the object tree. You have **Fields**, **Properties**, **Methods** and **Events** as the available choices.

Show

This group controls the level of information that is displayed in the object tree.

Categories

The **Show member categories** option will group all events together under an Events parent node in the Objects panel. Properties, methods and fields are also grouped together under their own parent group node.

The **Show base class** *categories* option will group all events, methods, fields and properties under a single parent node.

You can also specify whether both the above groupings, base class categories and member categories, will only occur if the total number of members exceeds a specified threshold.

The **Show Object members in base category** option determines whether the members of the standard base classes, *System.Object*, *System.__ComObject* and *System.MarshalByRefObject* are always shown for an object in the Objects view.

Component Inspector - Type Handler

The Component Inspector has its own special handlers for **IEnumerator**, **IList** and **EventHandlerList** types that display the contained objects in a more user friendly way. The Type Handler options panel allows you to enable or disable the special handling of these types.

Text Editor Options

To personalize your editing environment, customize the appearance, cursor behavior and other items use the **Text Editor Options** dialog.

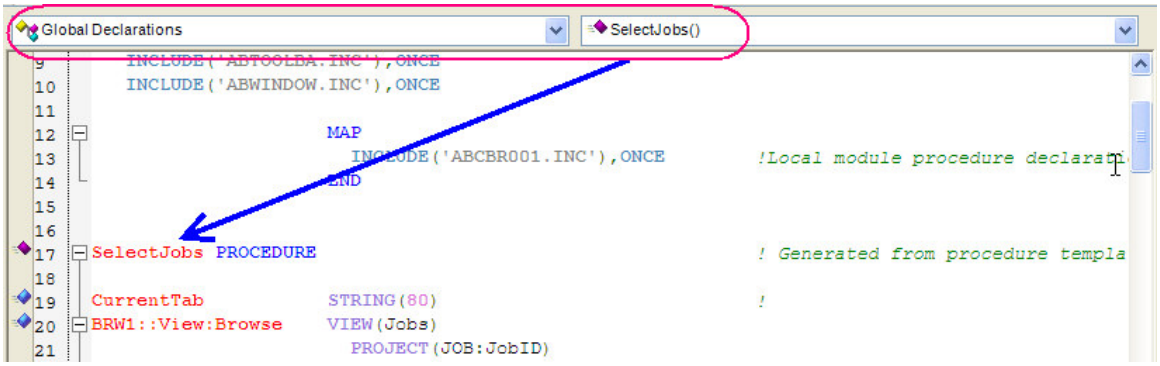
To view the dialog, choose **Tools ▸Options**. From the main IDE menu. Select the *Text Editor* tree item, which reveals the following set of sub-options.

General

Enable doublebuffering
On some machine configurations, the drawing of multi-colored text and other items can sometime cause flicker, due to the multiple passes that must be performed. Check this box to enable double buffering, which draws each line of text into an off-screen buffer before displaying it visually. This can have an effect of slower loading in some older machines, but doublebuffering in most cases is recommended.

Enable folding
Check this box to enable the folding of all data and code structures. This feature is handy for consolidating complex code sections, but also has a disadvantage of losing readability with more complex nested structures. Folding in most cases is recommended.

Show Quick ClassBrowser Panel
Check this box to display a special panel that allows you to quickly jump to selected areas of the text file.



The drop list on the left lists all data structures, like files and queues. The drop list on the right stores class structures used in this text file

Font
The Font settings allow you to control the appearance of text displayed in the Text Editor. A sample window is provided to view the font and other options that you have selected.

Markers and Rulers

The Markers and Rulers section controls the majority of the Text Editor's visual elements:

Show Horizontal Ruler
This option places a ruler at the top of the Editor window.

Show column ruler
Places a vertical line marker at whatever column you specify.

Line Marker Style
This option controls how your active line is displayed. Select Full Row for a special color indicator on the entire line.

Show line numbers
Activate this option to show line numbers on the left side of the editor window.

Underline errors
Activating this option will trigger an underlining of any errors produced by the compiler.

Highlight matching bracket
This option locates and highlights matching brackets whenever any bracket is active. This option is valid for ( ) [ ] { }

Show invalid lines
Activate this option to show invalid lines detected by the type of configuration that you have loaded. Using the "Clarion" highlighting scheme (the default), no lines are considered invalid.

Show EOL markers
Activate this option to show the "end of line" markers. These markers are identified by the special paragraph mark "¶"

Show spaces
Activate this option to show space markers. These markers are identified by standard "dot" markers.

Show tabs
Activate this option to show tab markers. These markers are identified by the special " **»** " character.

Behavior

The Behavior dialog window controls a wide spectrum of options that greatly affects your productivity in the Text editor.

Tab size
Enter the tab size in character size increments. Pressing the TAB key moves the caret (the place where something is to be inserted in a line) the specified number of characters.

Indentation size
Enter the indentation length in character size increments. Pressing the ENTER key indents the caret the specified number of characters.

Indentation (CTRL + I)

indentation functionality is divided into 2 essential parts:

- Indentation of the existing text (Ctrl + I) or formatting after pressing the Enter key.
- Cursor positioning on a new line.

There are three modes of indentation available in the Text Editor:

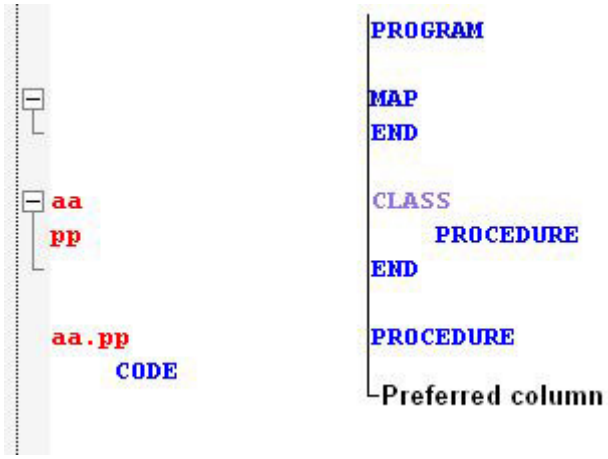
None
Set this to turn off all indentation effects. Existing text is not indented neither automatically nor by the CTRL+I hot key. Cursor is always placed to the first column on a new line.

Automatic
With Indentation set to Automatic ,the Text Editor keeps a running indent. When you press the Enter key, spaces and tabs are inserted to line up the insert point under the start of the previous line. In this mode indentation is set to the same value as on the previous line. For example, if the text on the previous line starts from the column 8, indented existing text will start from the column 8 (or the cursor on a new line will be placed to the column 8). If you try to indent the entire text in the document (CTRL+A, CTRL+I), all lines will start from column 1.

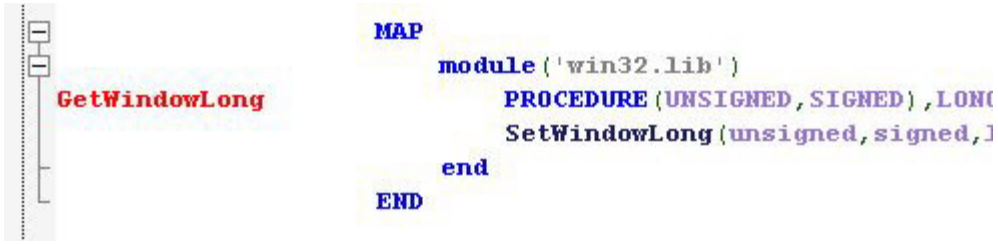
Smart
After a keyword statement, the next line is indented by the **tab size** set (see above). After certain keywords (break, return etc.) the next line is "non-indented".

Other rules for Smart include

- The **Preferred column** number (found in the Clarion Specific Options section) identifies a column where a keyword will be placed if it is not located already in some code block. In other words, if the keyword should not be indented relative to the parent it will be placed in the preferred column. The default value is column 21.
- Statements are indented from the CODE keyword position (default = on). CODE, DATA and INCLUDE (and USING and INLINE in Clarion#) keywords have one indent.
- END is indented to the same column as a keyword in the line that opens a code block:
- Keywords inside a code block have one indent from a parent keyword. For example, in the *pp* PROCEDURE:



- If a procedure inside a MAP (MODULE) is written without the PROCEDURE keyword, it will be indented:



- Any expression that ends with a colon ":" is treated as a label in the CODE section (default = off, where colons are treated as an actual procedure call)
- A line is indented after the Enter key is pressed at the end of the line (The default is on).
- You can automatically indent several lines of pasted text (The default is off).
- Source comments may also be indented (The default is off).
- If previous line has line continuation character '|' the next line will have one indent from the previous one.
- In the code definition section (prior to the CODE statement) the cursor is placed on column 1 on a new line. In the CODE section, the cursor is indented according to written statements.

Finally, if you are transferring code from another developer, and there are different **Tab Size** or **Convert Tabs to Spaces** options or the code is formatted manually, you may see the erratic behavior when using Smart indentation mode. This is due to the line(s) formatted according to your settings but all other lines remaining unchanged. To solve this issue you have several options:

1) Use the same tab size and indentation mode settings as the other developer.
2) Reformat the code before changing it (Ctrl+A, Ctrl+I).
3) Uncheck the **Enable entered line formatting** option so that the line will not be reformatted after pressing the Enter key.
4) Change indentation mode to *None* or *Automatic.*

See the Clarion Specific options dialog to control the options described above.

Convert Tabs to Spaces
When this option is active, all tab characters are auto converted to space characters when the document is opened.

Can move caret behind EOL
Checking this option allows you to move the caret marker beyond the EOL marker.

Automatic template insertion
This feature is not applicable to Clarion 7 at this time.

Auto insert curly braces
This feature is not applicable to Clarion 7 at this time.

Hide mouse cursor while typing
Check this option to hide the mouse cursor while typing. The mouse cursor will reappear when the mouse is subsequently moved.

Cut or Copy entire line when nothing is selected.
Activating (checking) this option will automatically move the entire line to the clipboard during any Cut or Copy action.

Mousewheel direction
This option is used to set the default scrolling direction of the mouse wheel.

Clarion specific options - General

The options contained in this section are unique to the Clarion IDE. The Smart Indentation options located in the Text Editor behavior section can be customized in more detail as follows:

Preferred column
Identifies the preferred column position of indentation, if the line to be indented is not based on a Parent keyword (e.g., CODE, IF, CASE, etc.).

Enable entered line formatting
Indents a line after the Enter key is pressed at the end of the line.

Indent statements from CODE
Allows indentation to be positioned relative to the location of the CODE statement.

Treat statement ends with colon as label
This option tells the editor to treat expressions that ends with a colon (**:**) as a statement label in the CODE section. If this option is off, expressions ending with a colon are treated as a procedure call. You need to have the caret position at the beginning of the target statement, and then press Enter. With option on, statement is moved to column one on the next line. With this option off, the statement is indented.

Indent comments
Indicates that any commented text will be treated as other non-commented text based on the current settings. For example, if the "pasting" option is active below, comments will be indented when pasting. If this option is off, comments will not be indented when pasting.

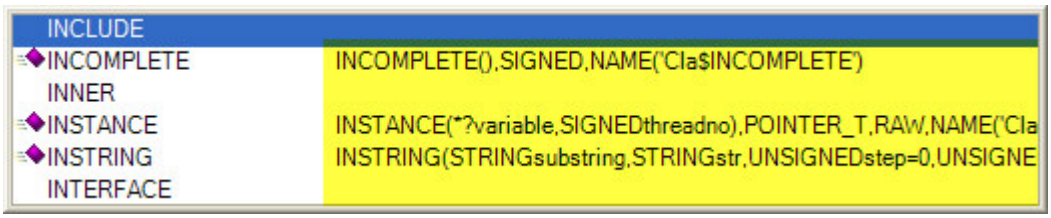
Format text after pasting several lines
Auto format (e.g., indent according to your settings) pasted text if several lines are pasted.

Clarion specific options - Clarion for Windows

Code Completion Options

Add code snippets to code completion and word completion lists
Check this option to place the IDE Code Snippets into the code and word completion lists.

Display declaration information in code completion list
Check this option to display the full declaration of the code statement, highlighted below:



There are also two resizing options available for the code completion window. Specify the **Number of rows in the completion list** to show, and also you have the option to **Remember completion list width**.

Complete options:

Keywords, Clarion attributes, and built-in data
Designate how to complete, either in upper case or lower case.

Other names
Designate how to complete all other names that are not keywords, attributes, or built-ins. Select from *As declared*. *upper case* or *lower case*.

Auto Insertion options:

Insert line continuation character after enter is pressed
When this option is active, a line continuation character will automatically be added on enter if the last character on the line does not meet the criteria entered below.

Exclude Characters
By default, line continuation characters are not inserted if the last character in the line is a letter, digit, or one of the characters entered as shown.

Clarion specific options - Clarion.Net

Code Completion Options

Add code snippets to code completion and word completion lists
Check this option to place the IDE Code Snippets into the code and word completion lists.

Display declaration information in code completion list
Check this option to display the full declaration of the code statement, highlighted below:

These declaration options only apply to code statements in the Clarion runtime libarary.

There are also two resizing options available for the code completion window. Specify the **Number of rows in the completion list** to show, and also you have the option to **Remember completion list width**.

Insert 'using' directive as:
This option allows you to choose preferred syntax for automatic using insertion. Select from three different styles (all are supported by the compiler).

Sort 'using' directives after adding new one
If enabled and you add a new 'using' by the right click menu command, all 'using' directives will be grouped to 2 groups (system and the others) and both groups will be sorted alphabetically.

Complete options:

Keywords, Clarion attributes, and built-in data
Designate how to complete, either in upper case or lower case.

Other names
Designate how to complete all other names that are not keywords, attributes, or built-ins. Select from *As declared*. *upper case* or *lower case*.

Auto Insertion options:

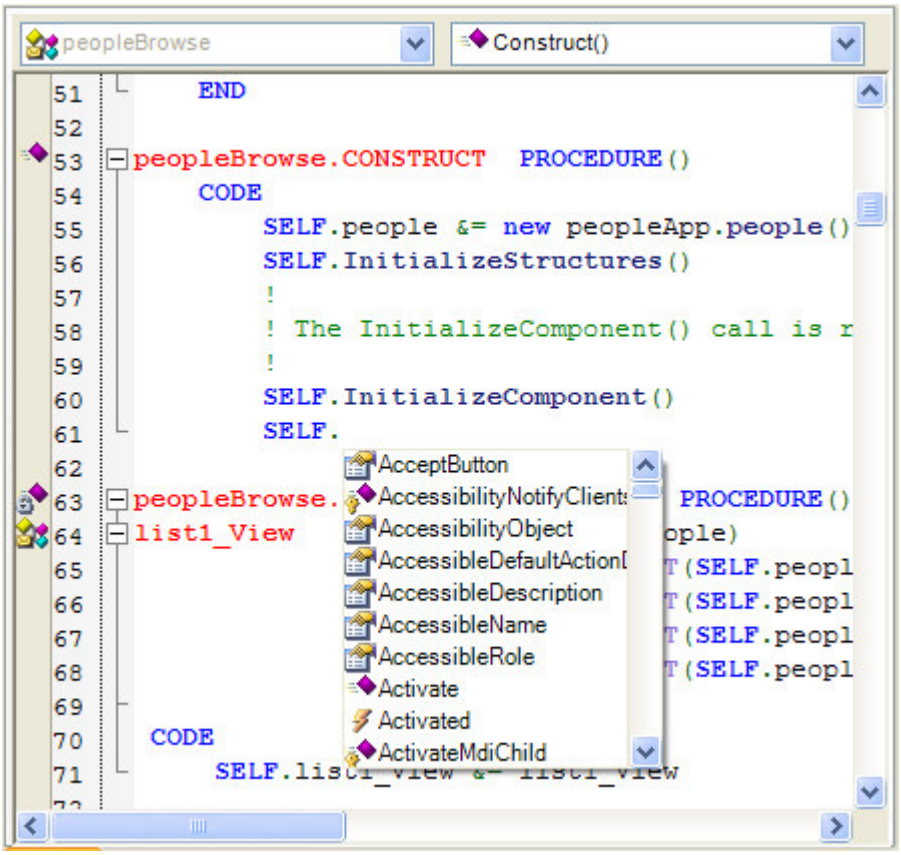
Insert line continuation character after enter is pressed
When this option is active, a line continuation character will automatically be added on enter if the last character on the line does not meet the criteria entered below.

Exclude Characters
By default, line continuation characters are not inserted if the last character in the line is a letter, digit, or one of the characters entered as shown.

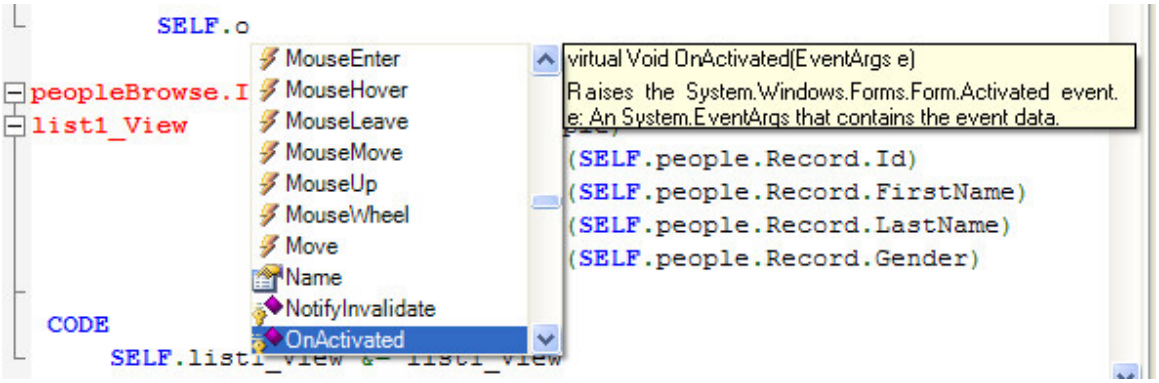
Code Completion

Clarion provides code completion as you type helping you to quickly find and enter an object's method, property or event.

Typing the dot character "." after an object's name will automatically bring up a list of possible methods, properties and events that are available for that object.



Typing the first few characters of the method, property or event will select the matching list item. Pressing the **Tab** key will complete the code by inserting the currently selected list item into the source code.



You can toggle the Code Completion feature here, on the IDE **Edit** Menu, IDE toolbar, or by hot key.

In addition, after certain keywords (like NEW and USING), pressing **CTRL + SPACE** will also display the code completion drop list. Enter **ALT** + **RightArrow** to see only the identifiers that start with typed expression.

Enable Code Completion
Check this box to activate the code completion feature.

Pre-select recently used members
Check this box to allocate a memory cache of recently used items. This will allow for quicker parsing in large projects. Set the number of items in the **Save** option, and press the **Clear Cache** button to remove all items from memory.

Show tooltip when moving mouse over expression
Check this box to activate the assisted tooltip when moving your mouse over any valid expression. Check the **Only in debug mode** box to only enable in Debug mode.

Narrow down completion list on typing
This option causes the typed word to act as a filtered locator for the code completion list.

Add new line after enter in case of fully typed word
If you press the ENTER key, and the word selected in the completion list equals the word in the typed text, a new line will be inserted. Otherwise, if the option is disabled, the ENTER key just closes the completion list.

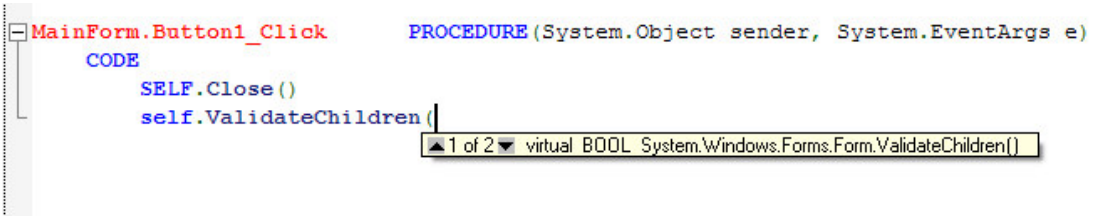
Complete word after 'insertion key' (space, dot, bracket, etc.)
If enabled and you press a dot after partially typed word, the word will be completed. For example, you type "sys" and a "System" is selected in the completion list, after pressing a dot you will have "System.". If the option is disabled, you will have "sys.". "Insertion key" is actually any character except letters, digits and underscore (and colon in Clarion).

Language Dependant Options

Show completion list after a character is typed
When active, the code completion window is automatically displayed as you type in any character that does not exist on column one (1).

Trigger code completion after keywords
Shows code completion after some of the language keywords. For example, in C# after you press SPACE after the "new", "as", "is", "override", etc. keywords, you will see the code completion window with suggestions (this works also if you enter an opening "(" after some keywords, like "for", "foreach", etc.). In Clarion/Clarion# there is only one keyword (DO) that triggers a completion list, and this displays a list of routines that can be called.

Show tooltip when writing method calls
When active, as the parameters of any method call is entered (by opening parenthesis), a tool tip pops up that offers the possible method parameters, if applicable. When this is active, you also have the option to **Re-open tooltip with better overload when pressing comma.** This checkbox reopens the tooltip with a more restricted set of choices, based on the parameters already entered. If the method is overloaded you will see all variants in the tooltip. But if you start writing the arguments of the method call, the number of possible variants is reduced so at the next comma the tooltip should be updated to show only overloads that receive typed arguments. This functionality should work in C# projects but at this time is not supported in Clarion.



XML Options

The Text Editor has built in support for XML (Extensible Markup Language). You can edit whole XML documents, or embed XML documentation in any Clarion project. The mechanics of XML Documentation is found in the Code Completion help topic.

Show attributes when folded

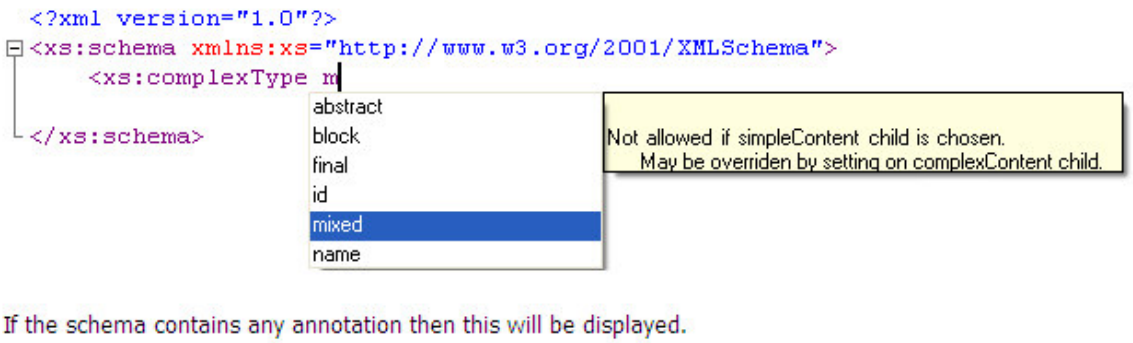
XML tags attributes can be folded. Check this option to display the XML attributes in the folded state. Example:

```
<name attribute="value">content</name>
```

If this option is active, the folded text will include the "attribute=value" text. If unchecked, only the <name> tag would be visible.

Show schema annotation

XML Completion is built into the Clarion IDE. When this option is active, you will see a display similar to the following:



If the schema contains any annotation then this will be displayed.

XML Schema

The list of active schemas used in the code completion of any XML document are listed here. To add new schemas, press the **Add** button. Only user-defined schemas can be removed from this list (built-ins are disabled)

The File Extensions list contains file extensions that are associated with the XML editing capabilities. If you need to add additional extensions, add them here.

Highlighting

The Clarion Text Editor provides several built-in highlighting schemes for a wide variety of languages. Each scheme is activated by the designated file extension.

To modify any existing scheme, highlight the desired scheme and press the **Copy to user-defined** button. From there, highlight in the right pane the scheme just copied, and press the **Modify** button. In the subsequent dialog, you can modify file extensions, keywords, color schemes, and rule sets.

Tools

External Tools

The Tools options contains setup and configuration options found in the commonly used accessory tools of the IDE.

Tools List

The elements in the Tools List are displayed in the top of the IDE **Tools** menu. As each item is selected, the following options are available:

Title
Enter the name to display on the IDE Tools Menu

Command
Enter the name of the program to execute. Use the ellipsis button to help you locate the correct program name and path.

Arguments
This tool accepts arguments in C# format. Use the button on the right of the entry to assist you in selecting the properly formatted argument type.

Working dir
Enter a working directory, or use the button on the right of the entry to assist you in selecting the properly formatted directory.

Prompt for arguments
Check this box to popup a window that allows you to enter arguments at execution time.

Use Output Window
Check this box to open a CMD shell for running WIN32 based programs.

Tools Options: Tools - Code Analysis

The Code Analysis option requires that you have FxCop installed. It is a feature supported for Clarion# only.

FxCop is a code analysis tool that checks .NET managed code assemblies for conformance to the Microsoft .NET Framework Design Guidelines. It uses reflection, MSIL parsing, and call graph analysis to inspect assemblies for more than 200 defects.

For more information, see the following link:

http://www.gotdotnet.com/Team/FxCop/

Tools Options: Tools - Code Coverage

The Code Coverage option requires that you have NCover installed. It is a feature supported for Clarion# projects only.

NCover is the open source code coverage tool for .NET.

NCover provides statistics about your code, telling you how many times each line of code was executed during a particular run of the application. The most common use of code coverage analysis is to provide a measurement of how thoroughly your unit tests exercise your code. After running your unit tests under NCover, you can easily pinpoint sections of code that are poorly covered and write unit tests for those portions. Code coverage measurement is a vital part of a healthy build environment.

For more information, see the following link:

http://ncover.org/site/

Tools Options: Tools - Help 2.0 Environment

The Clarion 7 and Clarion# environment includes support for MSHELP 2 Help Collections. The core information regarding Clarion and Clarion# is found in this HTML Help file, and MSHELP 2 is not required.

In this dialog, select an available help collection to access in your development environment.

Windows Forms Designer - General

The Windows Forms Designer is used in designing .NET applications. Select form the following options:

Use Smart Tags
Smart Tags are shortcuts for certain properties and events. For example, given a List Box control, selecting the "Edit items" tag is much the same to press the ellipsis (...) button on property "Items". The "Edit items" tag is the Smart Tag.

Enable in-place editing of ToolStrips
Toolstrips are special controls are toolbars that can host menus, controls, and user controls in your Windows Forms applications

Automatically open Smart Tags
The Windows Forms designer allows developers to perform common tasks and edit common properties directly from the design surface.

When a control is selected or when the mouse hovers over a control’s surface, a smart tag anchor is shown. Clicking this opens a panel containing common control properties and tasks that developers can perform without writing a single line of code. The smart tag allows the creation and selection of a data source without leaving the designer.

Use optimized code generation
Optimizing code generation can result in a more efficient link, but may disable some debugger features.

Insert "TODO" comment in new event handlers
An event handler is a procedure in your code that determines what actions are performed when an event occurs, such as when the user clicks a button or a message queue receives a message. When an event is raised, the event handler or handlers that receive the event are executed. Events can be assigned to multiple handlers, and the methods that handle particular events can be changed dynamically. This option places a "TODO" comment in event handlers that have not be created yet.

Activate the designer after file opening
When this setting is on, the Windows Forms Designer will automatically open a valid source file into the Designer from the Properties Pad. This is similar to clicking on the Design tab. This option should be turned off for larger projects in order to increase the initial project load time.

Sort properties alphabetically
Check this option to sort the properties dialog alphabetically by default.

Windows Forms Designer - Grid Options

The Clarion 7 Structure Designer offers two ways to position and align your controls, using either **Lines** or the **Grid**.

Snap to Grid
Determines whether the Window Designer will snap the controls to the grid. The resizing and movement of controls on the designer are constrained to the **Grid Width** and **Grid Height** when this feature is turned on. Having **Snap to Grid** turned on makes it easier to line up the various aspects of the user interface precisely, but limits the freedom with which one can place controls.

Snap Lines
Aligning controls relative to each other in a form has always been a challenge. Snap lines are a new feature that give you visual suggestions to help you more easily place and size controls relative to each other. Snap lines appear when controls are dragged from the Toolbox or moved around the Window designer area. The advantage of snap lines is that you are given guidance on not only alignment, but also proximity. Not only can you line up the bottom border of a Prompt and Entry control, but you can also manage the distance between them to create a consistent aesthetically pleasing look.

Grid Width
Enter the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on.

Grid Height
Enter the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

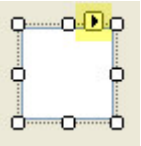
Show Grid
Check this box to automatically display the grid in the Designer work area.

Snap to Grid
Check this box to turn grid snap on; uncheck it to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

WINDOW Structure Designer - General

General Options:

Use Smart Tags
Smart Tags are shortcut button controls used at design time for certain properties and events. For example, given a List Box control, selecting the Smart Tag button gives access to all links seen in the Properties window.



Automatically open Smart Tags
Check this box to automatically enable the Smart Tags on open. If a control is selected, checking this will display the Smart Tags popup menu as the control is selected.

Enable SuppressTransparency by default
Check this box to enable the design time **SuppressTransparency** property. The effect of this property is to allow the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. It has no effect on the actual runtime window display.

Enable UseVisualStyles by default
Use this option to control the display of Visual Styles for all valid controls in design mode (controls appear as if the XP manifest was active).

Select Toolbox when Designer opened
Check this box to automatically open (if not already opened) and select the Windows Control Toolbox when the Window Designer is opened.

Ask on Closing Designer
This option when checked will ALWAYS prompt you to save a window on exit, even if no changes were made. If this option is unchecked, a window will always auto-save on exit, and you will not be prompted to Save or Cancel.

Property pad settings:

Sort properties alphabetically
Check this option to sort all properties display in the Properties window in alphabetical order. The alternative to this is a display by category.

Designer Area Options:

Show Command Toolbar
The Command Toolbar displays all alignment options, and other special options, including **Tab Order, Print Preview, Switch SuppressTransparency,** and **Switch UseVisualStyles**

Show Properties Toolbar
Displays common properties for any selected control, including **Use Variable**, **Text** and **Font** attributes.

Always Show Scrollbars
If this option is off scrollbars are displayed only if the size of the WINDOW is greater than the display area. If this option is on scrollbars are always displayed regardless of the size of the WINDOW.

WINDOW Structure Designer - Grid Options

The Clarion 7 Structure Designer offers two ways to position and align your controls, using either **Lines** or the **Grid**.

Snap to Grid
Determines whether the Window Designer will snap the controls to the grid. The resizing and movement of controls on the designer are constrained to the **Grid Width** and **Grid Height** when this feature is turned on. Having **Snap to Grid** turned on makes it easier to line up the various aspects of the user interface precisely, but limits the freedom with which one can place controls.

Snap Lines
Aligning controls relative to each other in a form has always been a challenge. Snap lines are a new feature that give you visual suggestions to help you more easily place and size controls relative to each other. Snap lines appear when controls are dragged from the Toolbox or moved around the Window designer area. The advantage of snap lines is that you are given guidance on not only alignment, but also proximity. Not only can you line up the bottom border of a Prompt and Entry control, but you can also manage the distance between them to create a consistent aesthetically pleasing look.

Grid Width
Enter the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on.

Grid Height
Enter the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

Show Grid
Check this box to automatically display the grid in the Designer work area.

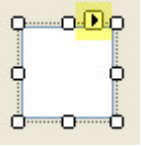
Snap to Grid
Check this box to turn grid snap on; uncheck it to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

REPORT Structure Designer - General

The options of the REPORT Structure Designer are identical to the WINDOW Structure Designer options discussed above.

General Options

Use Smart Tags
Smart Tags are shortcut button controls used at design time for certain properties and events. For example, given a List Box control, selecting the Smart Tag button gives access to all links seen in the Properties window.



Automatically open Smart Tags
Check this box to automatically enable the Smart Tags on open. If a control is selected, checking this will display the Smart Tags popup menu as the control is selected.

Enable SuppressTransparency by default
Check this box to enable the design time **SuppressTransparency** property. The effect of this property is to allow the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. It has no effect on the actual runtime window display.

Select Toolbox when Designer opened
Check this box to automatically open (if not already opened) and select the Report Control Toolbox when the Report Designer is opened.

Ask on Closing Designer
This option when checked will ALWAYS prompt you to save a window on exit, even if no changes were made. If this option is unchecked, a window will always auto-save on exit, and you will not be prompted to Save or Cancel.

Property pad settings

Sort properties alphabetically
Check this option to sort all properties display in the Properties window in alphabetical order. The alternative to this is a display by category.

REPORT Structure Designer - Grid Options

The Clarion 7 Structure Designer offers two ways to position and align your controls, using either **Lines** or the **Grid**.

Snap to Grid
Determines whether the Window Designer will snap the controls to the grid. The resizing and movement of controls on the designer are constrained to the **Grid Width** and **Grid Height** when this feature is turned on. Having **Snap to Grid** turned on makes it easier to line up the various aspects of the user interface precisely, but limits the freedom with which one can place controls.

Snap Lines
Aligning controls relative to each other in a form has always been a challenge. Snap lines are a new feature that give you visual suggestions to help you more easily place and size controls relative to each other. Snap lines appear when controls are dragged from the Toolbox or moved around the Window designer area. The advantage of snap lines is that you are given guidance on not only alignment, but also proximity. Not only can you line up the bottom border of a Prompt and Entry control, but you can also manage the distance between them to create a consistent aesthetically pleasing look.

Grid Width
Enter the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on.

Grid Height
Enter the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

Show Grid
Check this box to automatically display the grid in the Designer work area.

Snap to Grid
Check this box to turn grid snap on; uncheck it to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

Device Tools - General

All configurable features in Clarion.NET Compact Framework are placed under the Device Tools node in the **Options** dialog.

Show device choices before deploying a device project
If this checkbox is checked, before any deployment there will be displayed a special dialog with device choices. If not checked , the dialog is not shown and the default device is used.

Show skin in CF Forms Designer
Turns off displaying of a skin in the CF Form designer.

Always run as a desktop application
Allows the user to run a CF application as an ordinary desktop one.

Always use the desktop debugger
Allows the user to debug a CF application as an ordinary desktop one.

Device Tools - Devices

This option allows you to configure any device of any installed CF SDK.

Select a Device and press the **Properties** button. From there, you can press the **Emulator Options** button on the resultant dialog, and configure its properties as needed.

Device Tools - Form Factors

This option allows the configuration of any form factor of any installed CF SDK.

Select a Device and press the **Properties** button. From there, you can modify specific device options as needed. Examples of these options are **Skin** type, **Design-Time Defaults**, and other **Skin Defaults**.

Clarion - Clarion for Windows

Tools Options: Clarion - Clarion for Windows - General

If you wish to improve the initial load time of your active project, turn off (uncheck) the **Enable project parsing** option.

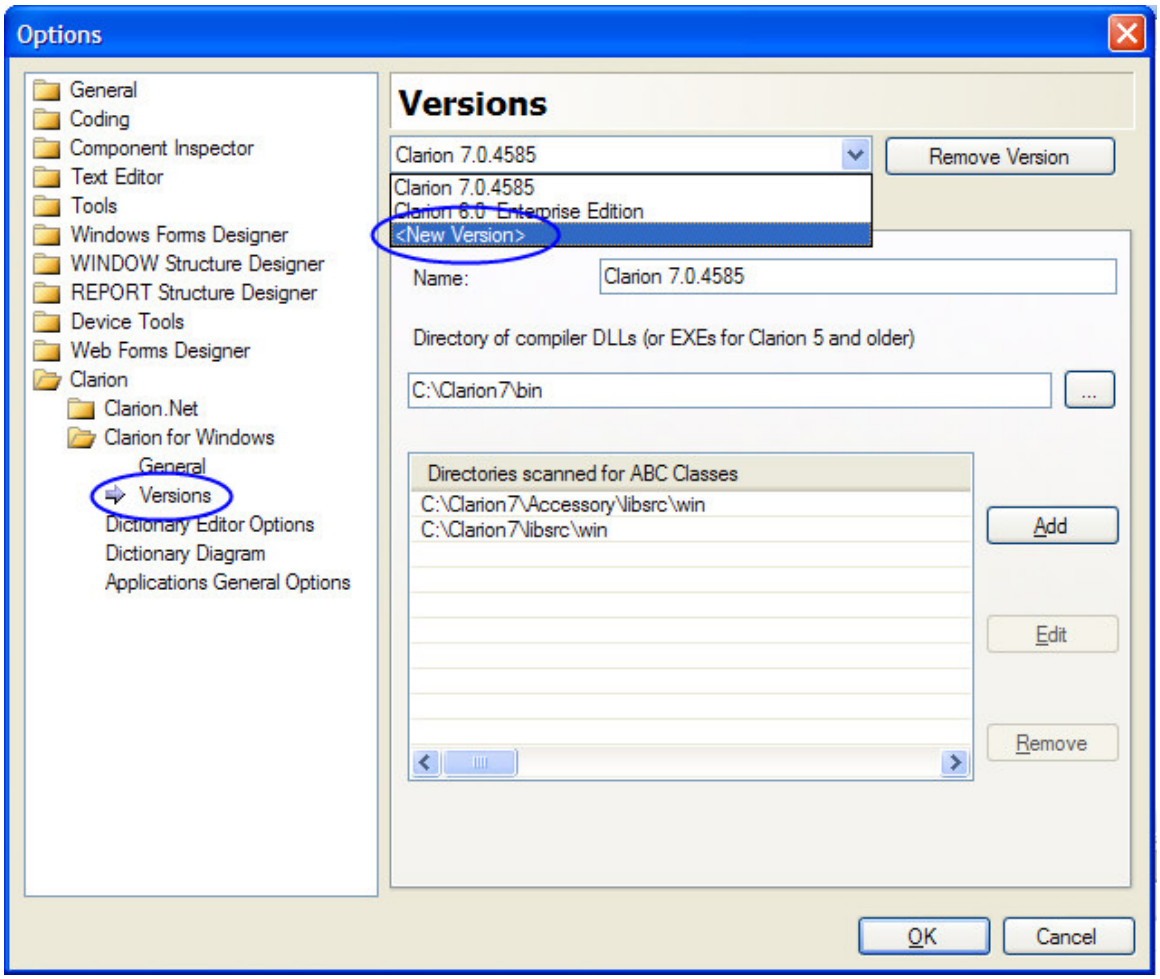
Turning this option off will also disable any code completion options that may be active.

Tools Options: Clarion - Clarion for Windows - Versions

The options in this dialog control the versions of Clarion that the IDE will register and recognize in the project system.

Use this dialog to verify versions, roots and redirection paths. The versions that you specify here will be available directly in the IDE Project Options.

Versions are detected by the IDE by reading the install entries in the Windows Initialization file (WIN.INI). If for any reason a version you have installed is not detected, you can manually add it here. You can also remove any version that you do not need to use in the IDE.



General

This tab displays the **name** of the current version selected, a path that points to the version's **compiler** DLLs (or EXE), and paths where the IDE will scan for ABC Classes.

If for some reason you do not see ABC headers being read and there are no ABC classes loaded in the IDE, you can add/change/remove the ABC path used by the install.

Redirection File

In Clarion 7 the redirection system now allows you to include other redirection files using the {INCLUDE} directive. This makes it much easier to use local redirection files to specify special paths for an application.

Clarion 7 also allows you to use older versions of Clarion to build your project. By default when you use an older version of Clarion, the INCLUDE directive is disabled. This is to maintain compatibility with the older redirection files.

In the Tools/Options/Clarion/Clarion for Windows/Versions dialog, if you select an older version of Clarion and go to the Redirection File tab you will see that the Can use the {include} directive is not enabled. If you enable this and change the name of the redirection file you can use the new include directive and not break the redirection files if you need to use the older IDE.

After changing the redirection name you will need to create a *newname.red* file in the bin directory of that version of Clarion with the line

```
{include %REDDIR%\oldname.red }
```

For example, if you change your Clarion 6 Enterprise Edition redirection file from *C60EE.RED* to *C600EE.C7.RED* then you will need to create in your Clarion 6 *bin* directory a file named *C600EE.C7.RED* with the line:

```
{include %REDDIR%\c60ee.red }
```

Then you can create local redirection files by right clicking on the project in the Solution Explorer and selecting "Create Redirection File in the project directory" allowing you to use the power of the Clarion 7 redirection system without stopping you using the older IDE.

Edit Macro Dialog

Macros are used to add or substitute a custom path for a default template or macro name.

Template (Macro)

Enter the redirection macro or template name that will be used in the redirection file.

Example:

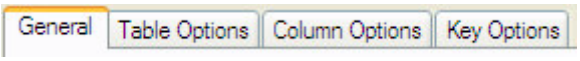
%ROOT%

Enter the name only here. In the redirection file the *%<template>%* syntax is used.

Description (Value)

Enter the valid path to substitute into the macro.

Dictionary Editor Options



Click on a TAB to see its help

You can customize the default dictionary settings in this dialog. To access the dialog, choose **Tools ▸Options ▸Clarion ▸Dictionary Editor Options**.

Note:

The display options in this dialog also affect the appearance of the Data/Tables Pad.

General

Initial Version Text

Sets the text to display to set the version of the dictionary. When you first create a dictionary it has an initial version with a name equal to what you set in that field. During the lifetime of the dictionary you can create new versions. Each version is a marker that is used in the Audit details of an object. If you go to the Audit tab of any object in the dictionary you will see a Version against the Created and Modified details. This indicates which version of the dictionary the object was created in and in with version it was last changed.

Add a Comment to the Dictionary when a new version is created

Check this option to automatically add a comment to the Dictionary every time the version is updated.

Version Comment

The text of the comment to add when auto adding using the option above.

Initial Freeze State

Defines the Initial Freeze State of the dictionary components as they are first imported.

Load Conversion Project

After a conversion program is generated this option sets how you can load the project. You can set this to *always* load the project, *never* load the project or *ask* each time a conversion program is generated. The default is to *ask*.

Enforce Clarion# name restrictions

By default this is on (checked). When checked, the dictionary editor will not allow you to create labels that are not valid in Clarion#. For example. *AS* is an illegal label, but legal in Clarion for Windows. If you do not want to worry about Clarion# restrictions you can uncheck this box.

If Clarion# restrictions are enabled and you load a dictionary into the editor that has a label that violates the Clarion# naming restrictions the editor will automatically rename the object to *ObjectType OldLabel*, where *ObjectType* is a Table, Alias,

Global, Pool, Key or Column. If the object is a Key or a Column with no external name, the external name is set to the old label.

For example, "AS STRING(10)" will be converted to

"Column_AS STRING(10),NAME('AS')

Any such change will appear in the Dictionary Changes pad.

Table Options

Default Driver — To select the default database driver for new tables in a dictionary, choose from the drop down list. For detailed descriptions of the drivers available, see *Database Drivers*.

Default THREAD Attribute — To specify new table definitions default to adding the THREAD attribute (setting aside a separate RECORD buffer for each procedure), check this box.

Default OEM Attribute — To specify new table definitions default to adding the OEM attribute (set International string support), check this box.

Sort Dictionary Tables — Select the *alphabetically* radio button to display the *Tables* list in alphabetical order, *date last modified* to display the tables in the order that they were last modified, and *date created* to display the *Tables* list in the order in which they were created.

Display Table Description — Check this box to display the table description in the Tables list.

Display Table Driver — Check this box to display the table driver in the Tables list.

Display Table Prefix — Check this box to display the table prefix in the Tables list.

Column Options

Display Column Description — Check this box to display the column description in the Columns list.

Display Column Type — Check this box to display the column's data type in the Columns list.

Display Column Picture — Check this box to display the column's display picture in the

	Columns list.
Display Column Prefix	Check this box to display the column's prefix in the Columns list.
Display Column Derivation	Check this box to display the column's derived column in the Columns list.

Key Options

Display Key Components	Check this box to display the Key components (fields) in the Keys list.
Display Key Description	Check this box to display the Key description in the Keys list.
Display Key Type	Check this box to display the Key Type in the Keys list.
Display Unique Flag	Check this box to display **Unique** if the Key is flagged as unique.
Display Primary Key Status	Check this box to display **Primary** if the Key is flagged as the Primary Key.
Display Other Key Attributes	Check this box to display the other attributes of the Key in the Keys list.
Display Key Prefix	Check this box to display the Key prefix in the Keys list.

Dictionary Diagrammer Options

This dialog allows you to fine tune and tailor many of the display and output options of the Dictionary Diagrammer tool when first opened.

Again, all of these settings in this dialog apply only to **NEW** diagrams. Once you saved a diagram, all of your settings are saved with it, including for example, the zoom factor.

General-Diagram

Initial Scale (Percentage)
Set the default Zoom percentage here. This setting will be active as the new Diagram is first opened.

Grid Options
Set the grid display techniques. You can specify that the grid is dotted (Points), a solid line (Lines), or neither (None).

General-Tables

Display Relationship Description
Check this box to display the full relationship description.

Display Table Grid Lines
The settings above here control the initial grid type and display settings. You can display points, lines or none in the Diagrammer.

Initial Size for Tables
Expanded displays all elements of the diagram. *Collapsed* displays a condensed list. *Name Only* displays the table name only.

Appearance

These settings control the initial colors displayed for the diagram and table backgrounds. The settings made here will ONLY apply to NEW diagrams that you create. To change existing attributes of a diagram, use the context popup menu provided in the diagrammer.

Diagram Background Color
Select from the drop list a background color to be applied to your new diagrams.

Table Font Color
Set the default font color to display for all textual elements in your new Diagram.

Table Background Color
Your tables can also have an initial background color. Select from the color drop list provided.

Arrow Color
You can control the initial color of your connecting arrows in all new Diagrams created.

Background Image
Press the ellipsis button select a background image for your diagrams. As you select the image, it will be displayed in the Options dialog.

Behavior

Double-Click toggles Expand/Collapse state
If set to ON, a dbl-click on any Table toggles the *Expand* or *Collapse* state of the Keys and Fields nodes. This can be useful when you have all tables collapsed and you want to quickly expand a particular table.

Rearrange Tables when Expanding or Collapsing
When you toggle the *Expand* or *Collapse* state of tables, the default behavior of the Diagrammer will rearrange the layout to either move tables further apart to prevent tables overlaying one another (if Expanding), or move the tables closer together to reduce excess whitespace (if Collapsing). If you already have your tables arranged exactly how you want them, you can set this checkbox to OFF (unchecked), and the tables will *not* be rearranged.

Printing

Print Diagram Background Color
Check this box to print the diagram background color

Print Diagram Background Image
Check this box to print the diagram's background image.

Print Table Fill Color
Check this box to allow the printing of the table background color. Otherwise, table colors will not be printed.

Application General Options

The options in this panel refer to settings in the IDE that affect all applications in regards to the project system. For options specific to the application internals see the Application Options topic.

Open application for Edit after Solution is loaded

Check this box to auto-open the Application Editor when a solution containing an application is detected. If a solution has multiple applications, this option is ignored.

Automatically add referenced Applications (.APP) to the Solution

Determines if dependent applications are automatically added to the solution during the conversion process. If set to OFF (unchecked), then after an application that has dependencies is opened for conversion, a dialog is displayed so you can decide if dependent applications should be added to the solution.

Cache application after edit

For more efficient handling of memory resources, uncheck this box to prevent the automatic caching (lazy loading) of your applications. Disabling this option is very useful and recommended for solutions with multiple applications.

If this option is ON, your application on close is still in memory and the associated dictionary is marked is read-only. You must close the solution to make the dictionary available for editing.

If this option is OFF then the dictionary is immediately available for editing when the application is closed.

Version Control System Support
Tools Options: Subversion

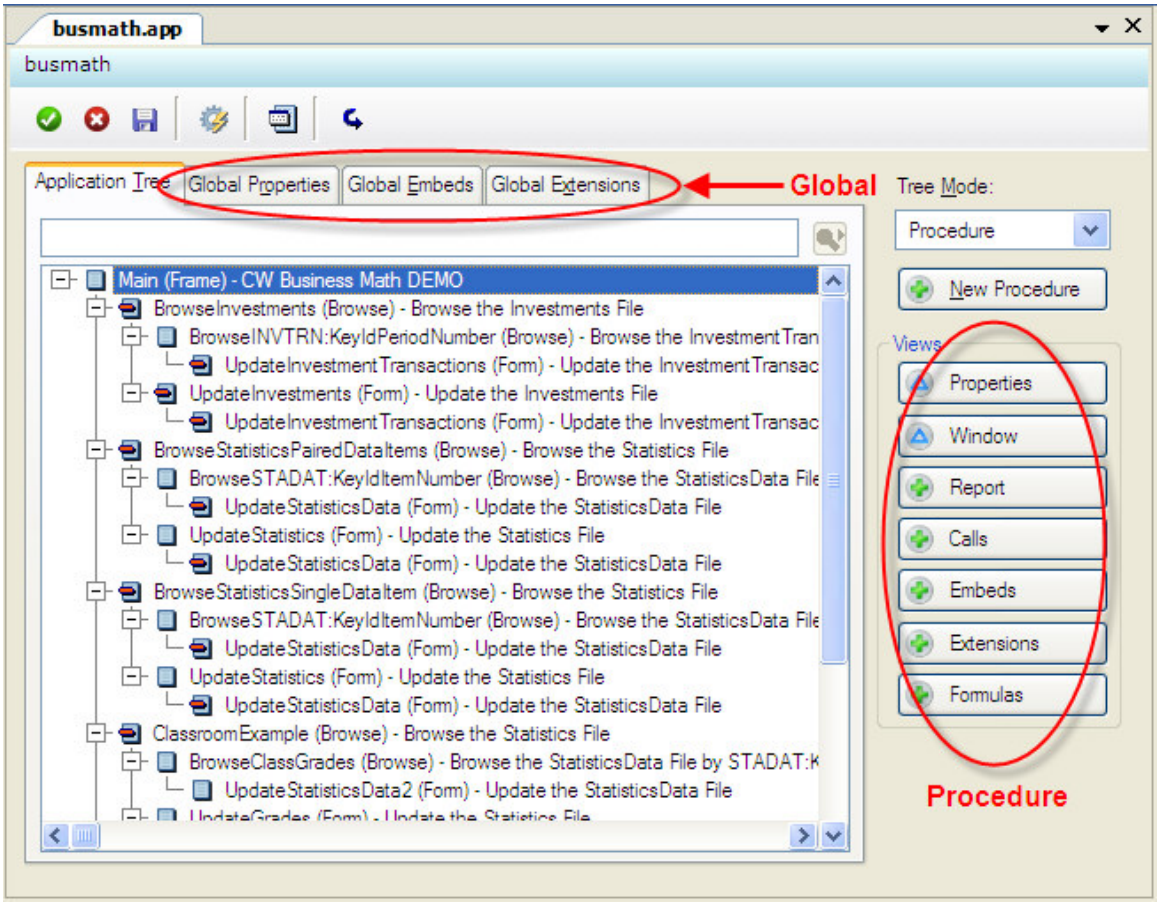
Subversion is a Version Control who's interface is easily integrated into Clarion 7 and Clarion#.
The options in this window control its basic default settings.
The GUI Interface to Subversion is called TortoiseSVN and can be found at the following link:
http://tortoisesvn.tigris.org/

More information regarding Subversion can be found at the following link:
http://subversion.tigris.org/

Application Generator

Application Tree

The core window of the Application Editor is called the *Application Tree*:



In this view, the **global** properties of the application are identified by the tab controls located above the procedure tree.

The buttons to the right are dialogs that apply to the selected procedure in the tree. More information regarding these button are described in more detail in the Procedures Properties dialog, where each of these buttons identify the special tab controls of the Procedure Properties dialog.

Note also that there is a special Application toolbar available, where buttons are provided for quick save or cancel of the application, and access to the Utility Templates and Source Editor.

Application Generator Menu Commands

The Application Generator generates your application's code based on the predefined templates you choose from the template registry.

This topic lists the IDE menu commands available from the Application Generator. Many dialogs also have Help buttons which you can press to view a help topic specifically about that dialog (the F1 key calls the same topic when the dialog is open).

Application Menu

Import From Application	Imports a procedure (or procedures) from another application (*.APP) file. First, select the application file to import from the *Select application to Import from* dialog. Next, choose procedures to import to the target application from the *Select for Export* dialog. You can select an item by double clicking on it, or. select the target procedure in the list and press the **Select** button. Items to be imported appear highlighted in red. Select additional items as needed. De-select an item by double clicking a previously selected item, or, select the target procedure in the list and press the **Clear** button. When ready to import, press the  to process, or the  to abort the process.  Both applications *must* use the same dictionary.
Import TXA (Text)	Imports the procedures defined in a .TXA (text) file. Create .TXA files with the Export Text or Selective Export commands.
Export Application to Text	Creates a .TXA (text) file from the current application.
Selective Export	Creates a .TXA (text) file containing only selected procedures.
Open Dictionary	Lets you view the current dictionary for the application.
Insert Module	Specifies a new MODULE for generated source code. You can also specify an external .LIB or .OBJ file to add to the project
Synchronize Application with the Dictionary	Applies the control attributes specified in the Data Dictionary to all the controls in the application. The attributes are applied as specified in the Synchronization tab of the Application Options

dialog.

Redistribute Procedures	Lets you change the number of procedures per module. Specify the new number in the Select Procedures per Module dialog. The Application Generator then redistributes the procedures among the modules, according to the new procedures per module number.
Repopulate Modules	Lets you change the number of procedures per module, but still keep related procedures together in the same module. Specify the new number in the Select Procedures per Module dialog. The Application Generator then redistributes the procedures among the modules, according to the new procedures per module number. Your application may execute slightly faster if you group procedures which commonly execute together in the same module
Renumber Modules	Renumbers the modules created by the Application Generator. This is useful for large projects from which modules have been deleted.
Delete Empty Modules	Removes empty generated source code modules from the project. This is useful for large projects from which procedures have been deleted.
Delete Empty Libraries	Removes empty external source code modules, .LIB files, and .OBJ files from the project. This is useful for large projects from which procedures have been deleted. The Delete Modules and Delete Libraries commands do not delete disk files, they simply remove any reference to the files from your application (.app) and project (.prj) files.

New Procedure	Adds a procedure not connected to the procedure tree.
Delete Procedure	Deletes the selected procedure, leaving it as a ToDo item in your Application Tree. To remove it completely, remove the reference or statement that calls the procedure.
Copy Procedure	Copies the selected procedure to a new procedure, which you name.
Synchronize Procedure with the Dictionary	Applies the control attributes specified in the Data Dictionary to all the controls in the selected procedure. The attributes are applied as specified in the Synchronization tab of the Application Options dialog.
Utility Template (CTRL + U)	Calls add-in utilities, including Clarion Wizards from the *Select Utility* dialog. See UTILITY for more information.

Popup (right-click) Menu

Properties Calls the selected procedure's Procedure Properties dialog.

Window Calls the **Window Designer** to visually design a window for the selected procedure.

Report Calls the **Report Designer** to visually design a report for the selected procedure.

Procedures Opens the Procedure **Calls** tab for the selected procedure.

Embeds Calls the **Embedded Source** dialog to manage embedded source code for the selected procedure.

Extensions Calls the **Extension and Control Templates** dialog to manage template generated code added to the selected procedure.

Formulas Calls the **Formulas** dialog or the **Formula Editor** dialog to manage formulas for the selected procedure.

Module Calls the **Text Editor** to edit or display the generated source for the selected procedure.

The source is as last *generated*. Any changes made directly to the generated source code are overwritten by subsequent source code generation.

When you right-click on any procedure and select **Module** from the pop up menu, the Editor stores certain information about the files you open to edit. This is a productivity aid; it remembers the line number you were last at, and so if a file is opened, positioned to a certain line number, closed, then reopened, you are now back at the line where you were last positioned.

Source Opens the Embeditor which lets you embed your own source code within the context of the surrounding generated code.

Synchronize Applies the control attributes specified in the Data Dictionary to all the controls in the selected procedure. The attributes are applied as specified in the **Synchronization** tab of the **Application Options** dialog.

Delete Deletes the selected procedure, leaving it as a ToDo item in your Application Tree. To remove it completely, remove the reference or statement that calls the procedure.

Application Generator Toolbar Buttons

The Application Generator displays a number of toolbar buttons to allow quick access to a variety of fundamental features. Most of these button functions are also accessible using standard menu or keyboard commands. Some of these buttons are also available in other areas of the Clarion IDE.

 **Save and Close**
Saves the active file to disk, and closes the application

 **Cancel**
Closes the application and aborts all changes in the active session.

 **Save**

Saves the active file to disk, and keeps the application opened.

 **Run Utility Template**

Opens the *Select Utility Template* dialog.

 **Invoke Embed Editor**

Opens the Embed Editor for the currently selected procedure.

 **Toggle Procedure Tree Mode**

Toggles between the *Module* and *Procedure* **Tree Mode View.**

Application Options Dialog

(Located in the IDE Tools Menu)

Click on a TAB above to see its help

The **Application Options** dialog allows you to specify default settings for each new application you create.

Application

Require Dictionary	This option specifies that each new application *must* have a data dictionary.
Default Dictionary	Names a data dictionary file as the default which appears in each new Application Properties Dialog . You can change to another dictionary before closing the dialog.
Display Repeated Functions	Check this box to have the Application Generator (Procedure tab only) provide full expansion of procedures called from more than one place in the Application Tree. Clear the box to provide full expansion of only the first instance--every other instance is marked "Expanded Above."
Procedures per Module	The **Procedures per Module** spin control specifies the number of procedures that the Application Generator writes to each source code module. **This can affect compile time when used with Conditional Generation turned on.** Specifying one procedure per module, for example, means that each successive compile rebuilds only those procedures changed since the last one, and no more. The down side to this is that it requires more disk space. **When using the ABC templates, we recommend 10-20 procedures per module.**
Application Wizard	Check this box to specify the default when creating a new application is to use the Application Wizard. You can override this choice when creating an application by checking or unchecking the Application Wizard box in the Application Properties dialog.
Procedure Wizards	Check this box to specify the default when creating a new procedure is to use the appropriate Procedure Wizard. You can override this choice when creating a procedure by checking or unchecking the **Procedure Wizard** box in the **Select Procedure Type** dialog.
Open Dictionary Files and Template Registry as Read-Only	Specifies file management options for multiple developer projects. When checked, Dictionary files and the registry are opened read-only.

Translate controls to control templates when populating — Check this box to have the Window Designer prompt you with a list of appropriate control templates whenever you place a control. We recommend this setting for new Clarion users.

Import name clash action — Specifies how the Application Generator handles procedure names from an imported application file that clash with procedure names already resident. Choose from the following:

Query on first clash — When the first clash is encountered, the Application Generator prompts for specific instructions on how to handle this clash and each subsequent clash. Choose from *Auto Rename*, *Replace All*, or *Prompt*.

- *Auto Rename* — Renames all imported procedures with name clashes by appending a sequence number to the imported procedure name.
- *Replace All* — Replaces all resident procedures with imported procedures of the same name.
- *Prompt* — Asks for specific instructions for each clash encountered. Choose from *Replace* or *Rename*.
- *Replace* — Replaces the resident procedure with the imported procedure of the same name.
- *Rename* — Prompts you to rename the imported procedure.

Ask for alternative — For all procedures with the same name, the Application Generator prompts you to rename each imported procedure.

Auto Rename — For all procedures with the same name, the Application Generator renames the imported procedure by appending a sequence number to the name.

Replace previous — For all procedures with the same name, the Application Generator replaces the resident procedures with the imported procedures.

Disable Column Prompts — Specifies that template-generated field-specific prompts will not display. This does not disable prompts created by Control Templates.

Sort Embeds Alphabetically — Check this box to show embed points in alphabetical order in the **Embedded Source** dialog. Clear the box to show the embed points in "logical" order (order of execution). See Embedded Source Code for more

information.

Action for Legacy embeds — Specify how the Application Generator handles legacy embed points. Legacy embed points are generally provided for backward compatibility among template chains. They allow newer template chains to conditionally support embed points from older template chains. See LEGACY in the *Template Guide*. Choose from:

Ignore all — Application Generator neither displays Legacy embed points at design time nor generates any code embedded therein. We recommend this setting to reduce "clutter" when developing new applications.

Show all and generate — Application Generator displays all Legacy embed points at design time and generates any code embedded therein. We generally do not recommend this setting; however, it can be useful for developers that are very comfortable with a particular template chain and its embed points.

Show filled and generate — Application Generator displays only filled Legacy embed points at design time and generates any code embedded therein. We recommend this setting for applications ported to a new template chain.

Edit embedded source errors in generated code — Check this box to allow you to edit embedded source errors in the generated source code instead of the standard Embeditor. The advantage of this is a logical view of all source code as generated. You can edit the source error and the application generator will not regenerate code until after the Make and Run is completed. This is useful during the testing phase of your development for faster prototyping.

Auto-Make Applications before compile — Check this box to trigger the Application Generator to generate source prior to the Build action if needed.

Auto start designer on pressing the Window or Report button. — Check this box to auto-load the target Designer when pressing the **Window** or **Report** button located on the Procedure Properties window. You can still view the Window or Report text by pressing the tabs in the Procedure Properties window. Closing the Designer will direct you to the appropriate Window or Report tab.

Move the MASK attribute from windows to controls — Check this box to allow the application convertor to automatically move the WINDOW MASK attribute to target ENTRY controls. If you plan to build the application in any version prior to 7, leave this check box blank.

Registry

Template Language code can be logically split among many files. Clarion uses the files to produce one logical template set for creating applications. The Registry Options are mainly for programmers who produce their own template files or make modifications to the default templates.

Re-register When changed — To automatically re-register your templates when the Application Generator detects a change, check the **Re-register When changed**

box. This defaults to "On."

Update Template Chain when edited	To automatically update the Template files when making a change in the Template Registry, check the **Update Template Chain when edited** box.
Regenerate Deleted Templates	To specify the Application Generator should re-generate the .TPL and .TPW files from TemplateRegistry.TRF, should the files be deleted, check the **Regenerate Deleted Templates** box.
#APPLICATION template	Select the default APPLICATION template from the drop-down list. The APPLICATION template controls source code generation. See *Template Overview* in the *Template Guide* for more information.

Generation

Conditional Generation	This check box specifies that only source code modules changed since the last make should be compiled.
Debug Generation	Specifies a text file for the Application Generator to log events to, and turns logging on and off. In case of a fatal error by the Application Generator, this log provides a trace to identify where the problem occurred. You can specify the file name in the **Debug Filename** box.  As you load (open) the application, you will see (DEBUG) in the title bar of the *Open Application* message window

.

Generation Message	Allows you to specify what displays during generation. Displaying messages increases generation time slightly, but offers more information on generation progress and more opportunities to cancel.

The choices are:

No Messages	Displays no messages.
Module Names	Displays the Module Name as it generates
Module and Procedure Names	Displays the Module Name as it generates and the Procedure Name as it generates
All Messages	Displays the Module Name as it generates, the Procedure Name as it generates, and each portion of the procedure as it generates.

Enable #ASSERT checking	Check this box to enforce heightened error checking during source code generation. This allows the Application Generator to identify certain template execution problems and notify you during source code generation. This slows the code generation process slightly. Clear the box for faster, but riskier source code generation. See #ASSERT for more information.
Use long file names for generated files	The default generation of application source files is the full application name followed by a three digit module number. If you would like to use the older 8.2 naming format, uncheck this box.
Use dot syntax for fields of structures	Used for the Clarion.NET Application Generator to generate field qualification syntax for all data structures instead of the older prefix:label format.
Create local maps	Check this box to generate a MAP structure for each source module that prototypes only the procedures referenced in the module. This results in faster compiles whenever you add new procedures or change procedure prototypes, because only the affected modules are recompiled. To generate accurate local maps, you must use the **Procedures** button in the **Procedure Properties** dialog to identify any procedures referenced in embedded source code.Clear this box to generate a single MAP for the PROGRAM module that prototypes all the program's procedures globally. This results in slower compile times whenever you add new procedures or change procedure prototypes, because the change to the PROGRAM module forces a recompile of all application source modules.
Enable embed commenting	Check this box to optimize automatic comment generation specified by the application templates.

Synchronization

This tab lets you specify how and when control attributes defined in your data dictionary are applied to your application's procedures and controls. **See Also: How to Synchronize your App and Dictionary .**

Synchronize Application when opened	Check this box to reapply data dictionary attributes each time your application is opened. Clear the box to apply the attributes only on your explicit command.
Synchronize Window definitions	Check this box to apply data dictionary attributes to WINDOW structures during application-wide synchronization. Clear the box to ignore WINDOW structures.
Synchronize Report definitions	Check this box to apply data dictionary attributes to REPORT structures during application-wide synchronization. Clear the box to ignore REPORT structures.
Update controls for variables	Check this box to apply memory variable control attributes to their associated controls. Clear the box to ignore controls associated

with memory variables.

Primary attributes only

Check this box to apply only the primary control attributes. Primary attributes are those attributes set directly in the Column Properties dialog. They include Column Name, Characters (length), Screen Picture, Prompt Text, Column Heading, Case (UPR, CAP), Typing Mode (INS, OVER), Flags (IMM, PASSWORD, READONLY), Justification, Initial Value, Help IDs, Messages, Tool Tips, and Validity Checks. Secondary attributes are those attributes set by pressing the **Properties** button on the **Window** and **Report** tabs of the **Field Properties** dialog.

Check the Primary attributes only box to speed up the synchronization process, especially if you synchronize each time you open the application.

Clear HELP, MSG, TIP if omitted in dictionary

Check this box to override control specific help attributes (set from the **Window Designer**) with blank help attributes from the data dictionary. Clear the box to retain control specific help attributes despite blank help attributes in the dictionary.

Allow control types to change

Check this box to apply new control types. For example, a SPIN may replace an ENTRY. The synchronizer does not change the number of controls on a window or report; therefore it does not change an OPTION to a LIST or vice versa. However, it does issue a warning when it encounters this situation.

Changing a control's type can result in "orphaned" embed code. For example a SPIN supports a NewSelection embed point, but an ENTRY does not. Orphaned embed code should be manually moved to an appropriate place.

Allow conversion from list to drop list

Check this box to allow a drop list to replace a list.

Clear all other attributes if omitted in dictionary

Check this box to override all control specific attributes (attributes set from the **Window Designer**) except HELP, TIP, and MSG with blank attributes from the dictionary. Clear the box to retain control specific attributes despite blank attributes in the dictionary. Clearing the box also enables the **More** button so you can set each attribute individually.

More

Press this button to elect, for each individual attribute, whether to override the control specific (**Window Designer**) attribute with a blank attribute from the dictionary or whether to keep the attribute despite a blank attribute in the dictionary. Attributes are **Font, Alert, Tally, Cursor, Key, Icon,** and **Colors.**

Dictionary can override size	Check this box to let data dictionary size attributes prevail over **Window Designer** size attributes. Control sizes can change when the height or width value is default and the control's text changes, or when an explicit height or width value in the dictionary varies from the control specific (**Window Designer**) height or width values.
Ignore Freeze attribute setting	Check this box to apply data dictionary attributes to controls with the #Freeze attribute. Clear the box to leave frozen controls alone. If you synchronize a single control, the synchronizer ignores the #Freeze. #Freeze is effective only when you synchronize multiple controls, i.e. an entire application or procedure, a WINDOW structure such as an OPTION or a GROUP, or a REPORT structure such as a DETAIL, BREAK, HEADER, or FOOTER.
Refreeze frozen control after synchronize	Check this box to" refreeze" the control after synchronizing it. Clear the box to "unfreeze" the control after synchronizing.
Update column formatting	Check this box to apply dictionary attributes to LIST FORMAT strings (i.e. justification). In other words, format list box columns according to data dictionary attributes. Clear this box to leave LISTs alone.
Update column headers	Select from the drop-down list to specify when List Box column headers are applied from the data dictionary. Choose from:
Always	The Application Generator always applies the dictionary column header, even if it is blank.
If present in dictionary	The Application Generator applies the dictionary column header only if it is non-blank, otherwise, the LISTs column header prevails.
Window and Dictionary	The Application Generator applies the dictionary column header only if *both* are non-blank, otherwise, the LISTs column header prevails. This lets a blank column header on a LIST prevail over a non-blank column header in the data dictionary.
Never	The Application Generator never applies the dictionary column header. The LIST's column header always prevails.
Display warning if could not synchronize	Check this box to display a warning dialog if warnings occur during synchronization. Warnings occur when controls change size, when there is a different number of radio buttons in the dictionary than on the window or when an OPTION is replaced with a different control (e.g. a drop-down list).
Add report entry when controls change size	Check this box to generate warnings when controls change size. Clear the box to suppress size change warnings.

Filename for report	Specify the file to hold the warning report. Clearing this box suppresses the report.

Embed Editor/OOP Embed Editor

This tab lets control how the Application Generator generates the temporary source file for the Embeditor. You can specify the text that delimits the embed points within the temporary source file. By customizing the text, you can make it easy to identify the embed points you want to edit.

Preceding Comment	Specify the text that marks the beginning of each embed point.
Include preceding comment	Check this box to generate a preceding comment. Clear the box to omit the preceding comment.
Prefix	Set the text generated before the embed point name.
Suffix	Set the text following the embed point name.
Following Comment	Specify the text that marks the end of each embed point.
Include following comment	Check this box to generate a trailing comment. Clear the box to omit any trailing comment.
Prefix	Set the text generated before the embed point name.
Suffix	Set the text following the embed point name.
Show priority levels	Check this box to show the embedded source priority within the Embeditor. The priority determines the sequence in which the Application Generator places multiple blocks of embedded code within a single embed point.
Edit errors in context	This box controls which edit mode to invoke when you edit embedded source code from the **Make Status** dialog. Check this box to open the Embeditor (equal to **Edit ▸Source**) to edit embedded source code. Clear the box to open the non-contextual embed editor (equal to **Edit ▸Embeds**).

Embed Tree Options

The Embed Tree is the relational list control that displays when selecting the Embeds menu item or button from any selected procedure.

The following settings allow you to fine tune the level of information displayed for all embed points:

Show PROCEDURE Keyword	Check this box to attach the word PROCEDURE to an embed point where appropriate.

Show VIRTUAL Keyword	If a method embed point is defined as a virtual method, check this box to attach the word VIRTUAL to an embed point where appropriate.
Show PROTECTED Keyword	If a method embed point is defined as one that is protected, check this box to attach the word PROTECTED to an embed point where appropriate.
Show Base Class	Check this box to display the base class name related to embed points (WindowManager, FileManager, etc.) where appropriate.
Show Object Description	Check this box to allow a class object's full description to be displayed.
Show Details	Check this box to allow a class object's full details to be displayed.
Color Entries	Check this box to reveal the color dialog described below.

Colors

Embed points can be colored with respect to what section of code it is created in. Modify your colors for DATA and CODE sections, existing VIRTUAL and PROTECTED methods, or new methods that you have created.

Application Properties Dialog

This dialog allows you to create a new application, or edit the "essential" information for an existing .APP file.

Application File Type a name for the .APP file. When opening the .APP file, the IDE makes the directory in which the .APP file resides the working directory.

Dictionary File Type the name of the data dictionary file (.DCT). You can press the ellipsis button (...) to locate the dictionary file using the Select Dictionary dialog. See How to Create a Data Dictionary for information on creating your application's data dictionary.

Note: The Application Generator does *not* require a data dictionary to generate an application, if you *uncheck* the **Require a dictionary** box in the Application Options dialog

First Procedure Type the name of the first procedure. This is usually called MAIN.

Destination Type Select Executable, Library or Dynamic Link Library.

Help File Optionally type the name of a Windows Help file (.HLP). The file must exist on disk, but it can be a "dummy" file. You can press the ellipsis button (...) to locate the help file using the Open File dialog.

Application Template The template controls code generation. You can select the default Clarion or ABC template, or choose a third party template set by pressing the ellipsis button (...), then choosing from the **Select Application Type** dialog.

To Do Template Accept the default ToDo(*template chain*) in the **To Do Template** field.

Application Wizard Check the **Application Wizard** box to use the wizard to create a complete application based on the selected dictionary and a few answers you specify. This option is only available when you are creating a new application.

Notes:

1. Create new .APP files *only* using the Clarion IDE. Do not copy a file (using the DOS command line, or File Manager) to a new file name, then open it in Clarion. This prevents the Application Generator from changing the internal names recorded in the file. If you need to copy and rename an .APP file, open it, then use the **File ▸Save As** command.
2. Application file names, besides being legal DOS names, must also be valid Clarion labels. The file name, 1MyApp.APP, for example, is illegal because it starts with a number instead of a character.

See Also:

How to Create a New Application File

BIND Fields and Procedures

The process of BIND ing a field, procedure or expression allows the entity be used in an expression string for either the EVALUATE procedure or a VIEW structure's FILTER attribute.

This template interface allows you to specify that a BIND is generated for the fields, procedures, and expressions you designate. The template interface generates the proper syntax for each of these entities.

On the **Fields** tab, you may insert, change or delete a field to be BINDed by selecting the **Developer Defined** tab, and selecting the appropriate button. You will be presented with an ellipsis that will allow you to select any variable defined within your application. The **Template Defined** tab displays any variables that have already been BINDed by other templates. You should always check this tab to verify a list of variables already BINDed.

On the **Procedures** tab, you may insert, change or delete a procedure to be BINDed by selecting the **Developer Defined** tab, and selecting the appropriate button. You will be presented with a drop list that will allow you to select any procedure name currently defined within your application. You can also add new procedure names not yet defined here. The **Template Defined** tab displays any procedures that have already been BINDed by other templates. You should always check this tab to verify a list of procedures already BINDed.

On the **Expressions** tab, you may insert, change or delete a name to be BINDed to an expression by selecting the appropriate button.

Name Enter a string value to use as the name to be used to identify the BINDed expression.

Expression An expression is a mathematical formula containing any valid combination of variables, functions, operators, and constants. Enter a valid expression to be BINDed. Example: (TODAY() – 1)

Press the "**E**" button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the **Expression** prompt.

Embedded Source Dialog

This dialog allows you to access Embed points from the *Procedure Properties* dialog.

By adding embedded source code to a procedure, you gain powerful customization capability. You can specify or create code to execute at any defined point in the source code. You can write your own code, or use a code template to write the code for you. The Application Generator adds your code to the code it generates, at precisely the point at which you specify.

This dialog lists all the available embed points, as defined by the procedure template.

To see your embedded source code in context, use the Embeditor instead (from this window, press the Source button. From the Application Tree, RIGHT-CLICK on the procedure and choose *Source*).

Edit Options

The Edit Toolbar Menu (and the edit buttons on the right) allow you to manipulate your embedded source code.

Save and Close	Save your editing session and closes the Embeds dialog
Cancel	Cancels your editing session and closes the Embeds dialog.
Save	Saves any changes and keep the Embeds window opened.
Actions	Opens the **Actions** dialog for the target template where approproiate.
Cut (CTRL+X)	Cuts the highlighted embedded source to the clipboard, allowing you to paste it into another embed point in this or any other procedure (including procedures in other applications).
Copy (CTRL+C)	Copies the highlighted embedded source to the clipboard, allowing you to paste it into another embed point in this or any other procedure (including procedures in other applications).
Paste (CTRL+V)	Pastes embedded code from the clipboard into the highlighted embed point.

View Options

This toolbar section allows you to adjust the display to show only what you want to see.

Show Filled Only	Show only filled embeds.
Show Priority Labels	Show template generated embed point labels so you can precisely interleave your code with template generated code.
Show Window Embeds	Available only when editing embeds for a control, this button allows you to expand the view to show embeds for the window.

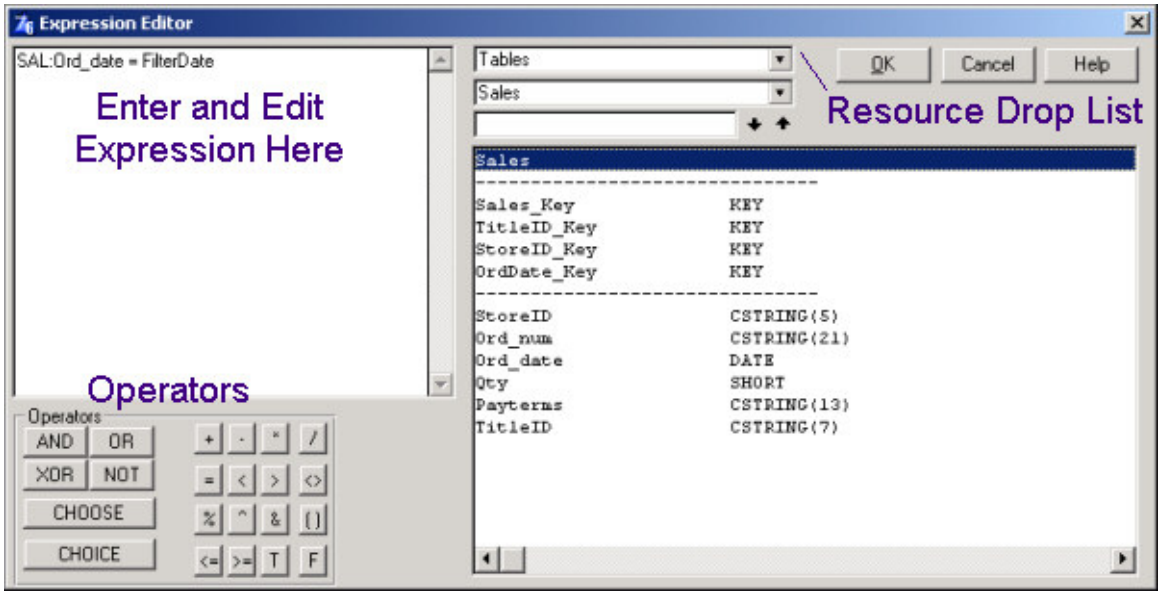
Show Legacy Embeds	Show Clarion 2.x embed points.  You may set the default for legacy embed points with the Application tab of the Application Options dialog. Choose **Tools ▸Application Options**.
Previous Filled (CTRL+P)	Moves the selection to previous filled embed point. It expands the tree as needed.
Next Filled (CTRL+N)	Moves the selection to next filled embed point. It expands the tree as needed.
Refresh	Refresh the Embeds Tree

Embed Buttons

Insert (INS)	Opens the Select Embed Type dialog, which allows you to add handwritten source code, call a procedure, and/or choose a code template.
Properties (ENTER)	Allows you to edit the embedded code. If it is hand written code, then the Text Editor appears. If it's a code template, the prompts dialog for the code template appears.
Delete (DEL)	Allows you to delete embedded code you previously added.
Move Up (CTRL+UP)	Moves the embedded code item up above another (modifying the Priority). Each executes in the order they appear at an embed point.
Move Down (CTRL+DOWN)	Moves the embedded code item down below another (modifying the Priority). Each executes in the order they appear at an embed point.
Priority	Sets the **Priority** for the embedded source. The **Priority** of each block within an embed point controls the execution sequence of the code relative to any other code in the same embed point. Lower priority numbers execute before higher priority numbers.  Moving an embed up or down changes its priority.
Filled button	Press this button to only show embed points that actually have had source code entered into them.
Source button	Press this button to call the Embeditor, allowing you to see your embedded source in context.

Expand All	Fully expand the embeds list.
Contract All	Fully contract the embeds list.
Expand Filled	Expand only the filled embeds.

Expression Editor Dialog



The Expression Editor is a built-in utility that helps you build all types of expressions.

(Chapter 10 of the Language Reference provides a detailed overview of Clarion language expressions)

Refer to the diagram shown above and the following prompts:

Enter and Edit Expression Here

The left window pane (a text box), is used to display the expression that you are constructing. You can enter an expression manually, or use the helper panes to the right to help you build your expression.

Resource Drop List

Use the drop list shown above to select from a variety of elements to use in your expression. This includes:

Clarion Functions	A list of available built-in Clarion functions
GlobalData	A list of the application's global data elements
ModuleData	A list of the current module's data elements
LocalData	A list of the current procedure's local data elements
Tables	A list of the tables defined in the current procedure
Procedures	A list of procedures called from the current procedure
Window Control	A list of window elements and their field equate labels

Based on the resource selected, additional prompts may appear to help you select additional elements. Locators are provided for all resources selected.

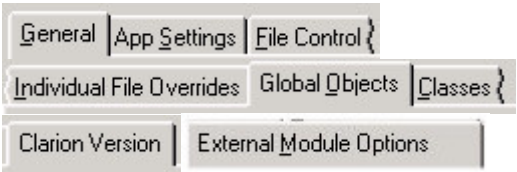
Operators

Provides buttons for inserting logical and bitwise operators into the expression. You can also type them in directly.

Global Data/Local Data/Module Data Dialogs

See: Data / Tables Pad

Global Properties



Click on a TAB to see its help

This dialog specifies application level options for file processing, .INI file support, plus lets you define global variables.

General

Program Author
Lets you add your own name, which is then added into the .APP file.

Default Icon
Press the ellipsis button to select an icon to use as the program icon. You may also specify a variable name here by prefacing the variable with an exclamation point (!). The templates implement the use of SYSTEM {PROP:Icon}.

Use Field Description as MSG() when MSG() is Blank
Check this box to tell the Application Generator to use field descriptions from the Field Properties dialog as the default MSG attribute.

Generate Template global data and ABC's as EXTERNAL
Adds the EXTERNAL attribute to the global variable declarations generated by the templates, and the DLL attribute to any CLASS declarations generated by the templates. This means your program relies on an external library to allocate memory for these variables and objects, and to export them so your program can access them.

You should add the EXTERNAL and DLL attributes to get the same effect for any global variables or classes you declare. See the *Language Reference* for more information on these attributes.

Note:

If you create a program that consists of more than one AppGen created DLL, you should check the Generate Global Data as EXTERNAL box for all the applications except one. See the *User's Guide--Development and Deployment Strategies*.

External Globals and ABC's Source Module
Specify whether the external library is dynamically or statically linked.

This sets the *flag* parameter of the DLL attribute for template generated class declarations. See the *Language Reference* for more information on the DLL attribute.

Generate Embed Comments
Check this box to generate identifying comments surrounding all embedded source code.

Non Volatile Storage Settings

The Clarion and ABC Templates non volatile storage support the use of .INI (standard windows initialization) files or the system registry. These are mediums that store information for an application between sessions.

One use for non volatile storage is to store the user's preferred window positions for the next session. Another use is to save program configuration settings between sessions. Clarion's procedure templates let you do both automatically when you enable non volatile storage support.

Location
Choose between *INI file* and *System Registry*

INI File Options:

.INI File to use
Specify whether to use the default.INI file name. By default, the application creates the .INI file with the same file name as your application. To use another name select the *Other* choice from the drop down list.

Other File Name
Specify another .INI file name other than the default. You may specify a variable, a full pathname, no path, or a path of (.\) to generate the INI file as shown below.

Other File Name	Resulting INI File Location
!variable	(The contents of the variable specified)
c:\programs\payroll.ini	c:programs\payroll.ini
payroll.ini	(windows system directory)\payroll.ini
.\payroll.ini	(current directory)\payroll.ini

Use App Directory
Check this box to direct your INI file to be located in your application directory.

Registry Options

App Registry Key
Enter the name of the key whose value is to be queried. This may contain a path separated by backslash '\' characters. Example: `SOFTWARE\SoftVelocity\Clarion7`

Registry Root
Use the drop list to select a valid registry root where your registry key will be located.

Disable Save/Restore Window Locations

Check this box if you would like to omit the last window size and position from being stored in the non-volatile storage source for all window structures in your application. You can locally enable only the windows that you want to maintain at the target procedure level.

Preserve
Once you've enabled INI support, the ABC Templates automatically save and restore the values of designated global variables. This provides a simple mechanism for saving and reapplying end user preferences or program configuration options. Press the Data button in the Global Properties dialog to define your global variables, and then press the Preserve button to designate selected variables to automatically save and restore.

In the Preserve dialog window, you have the standard buttons available **Insert**, **Delete**, or examine and modify (**Properties**) the preserved variable. You can also move the variables up and down in the list.

Enable Run-Time Translation
Generates code to translate window text based on the translation strings defined by default in the ABUTIL.TRN file. See Translator Class for more information.

Enable Fuzzy Matching
Check this box to generate the necessary objects to support the BrowseFuzzyMatching control template.

Fuzzy Matching Options

Ignore Case
Check this box to enable all fuzzy searches in your application to be case insensitive. This can be overridden individually. See *FuzzyClass- SetOption* method for more information.

Word Only
Check this box to enable all fuzzy searches in your application to search on whole words only. This can be overridden individually. See *FuzzyClass- SetOption* method for more information.

App Settings

Enable Window Frame Dragging
Check this box to disable the "Show Contents While Dragging Window" Windows option. This functionality is known by Microsoft to cause Windows internal messaging system problems which may lead to GPFs. We recommend that the **Enable Window Frame Dragging** remain checked.

XP/Vista Manifest

Press this button to access manifest options for Windows XP and Vista. Support for VISTA Execution Levels and UI Access is included.

Extended UI

Press this button to access the Extended UI interface. As an added feature to the initial release of Clarion 7, Clarion 6 includes this special template interface designed to add powerful XP styles easily to your existing applications that you are planning to migrate to Clarion 7.

Provide visual indicators on control with focus
This check box refers to certain input controls that will change color when you select them. Check this box to enable the **Set Visual Indicators** button, which allows you to designate the controls and colors to use with this feature.

Field Navigation

Use ENTER Instead Tab
The TAB key is the standard key press in Windows for moving from control to control. If you wish to replace this with the ENTER key (the DOS standard), check this box. In addition, the DownKey/EnterKey now works as the TabKey, and the UpKey now works as ShiftTab for moving to the previous control.

The ENTRY, SPIN, DROPCOMBO & COMBO controls use the standard remapping described above.

For the LIST & DROPLIST controls: The EnterKey works as the TabKey. The setting to enable the RightKey to work as the TabKey and the LeftKey to work as ShiftTab is local to the procedure (Disable Left/Right Key on List controls).

For BUTTON controls, the DownKey and RightKey works as the TabKey, and the UpKey and LeftKey work as ShiftTab

For RADIO controls, the EnterKey works as the TabKey.

For CHECK controls: The EnterKey,DownKey and RightKey work as the TabKey, and the UpKey and LeftKey work as the ShiftTab.

Exclude Controls
Press this button to designate special controls that will NOT use the ENTER key for navigation.

Default Action
Choose *Enable* to turn on Field Navigation for all procedures by default. Choose *Disable* to turn off Field Navigation by default. This settings can be overridden on the procedure level.

Choose *Runtime* to enable or disable the Field Navigation feature based on the **Default Value** expression.

Default Value
Enter an expression to use to evaluate at runtime whether or not Field Navigation will be available. Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the appropriate prompt.

Report Preview Mode (Clarion chain only)
Select the type of Print Preview you wish to use in your applications. Click on **ABC Class** to use the Preview Class used in the ABC template chain, or click **Procedure** to use a default preview procedure found in the Clarion template chain.

If **Procedure** is selected, accept the **Default Report Preview** procedure used by the Clarion template chain, or enter an alternate procedure name (one that you have modified, or obtained from a third-party source.)

Enable Auto Size BrowseBox Columns
Check this box to enable the auto resize column feature. At runtime, double-click on the right line of any column and it will automatically resize to fit the data contents. You can disable this feature in the local Browse procedure settings.

Enable List Format Manager
Check this box to enable the List Format Manager button.

Browse Active Invisible (Clarion Templates Only)
The "**Active Invisible"** property indicates whether to fill or refill the browse queue when the browse LIST is "invisible" because it is on a non-selected TAB or is otherwise hidden. Check this box to refill the queue when the LIST is invisible; and uncheck to suppress the refill.

Turning off Active Invisible improves performance for procedures with "invisible" browse lists; however, buffer contents for the invisible browse list are not current and should not be relied upon.

Enable Rebase (DLL Applications Only)
Check this box to enable the Image Base Memory Address button, which provides rebasing options for this DLL target application.

File Control

The **File Control** dialog lets you override some of the settings in your dictionary, as well as define how procedures will access files. You can specify file attributes for all files, or individually.

Generate all file declarations
Generates all file declarations in the dictionary, even if not specified in any procedure's Data / Tables Pad.

When done with a file (Clarion templates only)
Specifies whether the application automatically closes each file when a procedure is finished.

One way in which you can design your application to be a "well-behaved" Windows application is not to hog system resources. You can return file handles not in use by checking this box.

Enclose RI code in transaction frame
Enables rollback of data if an update fails.

If all files in a relation chain use the same file system, and the file system supports transaction framing, and you do not want transaction framing around the RI code, you must clear the check box here, and choose No for *each* file on the Individual File Overrides tab.

Issue template warning if LOGOUT() not allowed (Clarion templates ONLY)
Enables a compile time warning when your data dictionary includes a file driver which does not support the LOGOUT() function.

You should clear this check box for drivers such as dBase III. **See Also:** Database Drivers.

Seconds for RECOVER
Specifies the number of seconds to wait before invoking the RECOVER process. This is applicable only to Clarion files.

File Attributes

Threaded
Specifies whether the application generator adds the THREAD attribute to FILE structures. THREAD is needed for MDI browse and form procedures to prevent record buffer conflicts when the end user changes focus from one thread to another.

Use File Setting
Sets the THREAD attribute according to the data dictionary.

All Threaded
Adds the THREAD attribute to each FILE.

None Threaded
Omits the THREAD attribute for each FILE.

Create
Specifies whether your application should allow the creation of a data file should it not exist. Adds the CREATE attribute to the FILE structure.

Use File Setting
Sets the CREATE attribute according to the data dictionary.

Create All
Adds the CREATE attribute to each FILE.

Create None
Omits the CREATE attribute for each FILE.

External
Specifies whether the application generator adds the EXTERNAL attribute to FILE structures. EXTERNAL specifies the memory for the FILE's record buffer is allocated by an external library.

None External
Omits the EXTERNAL attribute from all file declarations.

All External
Adds the EXTERNAL attribute to all file declarations *and* lets you specify the **Declaring Module** and whether **All files are declared in another .APP.**

When using EXTERNAL to declare a FILE shared by multiple libraries (.LIBs, or .DLLs and .EXE), only one library should define the FILE without the EXTERNAL attribute. This ensures that there is only one record buffer allocated for the FILE and all the libraries and the .EXE will reference the same memory when referring to data elements from that FILE.

Generate File Declarations in Modules
This checkbox is valid (enabled) when global data and file declarations are local to the application (EXE/DLL target with no external files).

When this check box is on (checked), file declarations are created in separate source files (similar to the ABC FileManager methods which are located in "base class" (BC) files). These source files use a FC*x*.CLW naming convention (e.g., *myappFC0.CLW* - the first five letters of the application name are prepended to the source file name). All file declarations in the main module are then declared with the EXTERNAL('') attribute.

This option also generates all defined *global* data in other source files as follows:

*myapp*GLO.CLW stores the global data as it is defined in the main PROGRAM module.
*myapp*GLF.CLW stores the same global data but with the EXTERNAL attribute added.

where *myapp* is the first five letters of the application name.

In the main program module, an INCLUDE for *myapp*GLF.CLW is generated. If any global data is marked as Generate Last in the data dictionary, these data elements are defined appropriately in the program's main module only.

The default value on how many files are declared in each GL*n*.CLW module is set by a global template symbol *%FilesPerFCModule* using a default value of 100 files per module.

This option is essential for large applications using more than 500 files, but any application can use this option to better organize file and global data declarations. Use the **Generate Last** option only for variables that require it.

Export all File Declarations
Checking this box tells the Application Generator to add the file information to the Export file. This is only available when the project's **Target Type** is a DLL and you specify **None External** (see above)**.**

Declaring Module
The filename (without extension) of the MEMBER module containing the FILE definition without the EXTERNAL attribute. If the FILE is defined in a PROGRAM module, leave this field blank.

All files are declared in another .APP
Checking this box tells the Application Generator that the files are declared in another APP (rather than hand code). Application Generator adds the EXTERNAL flag on the File:Open flag (that controls CheckOpen), ensuring that files are opened and closed at the right time, thereby preserving the integrity of the file data buffers.

Enable Triggers Support (CLARION Template Chain Only)
Check this box to enable the Dictionary Triggers support that is part of the ABC Class FileManager. See the topic on Dictionary Triggers for more information.

File Access

File Open Mode
Specifies how your application opens files.

Open
Opens files as Read/Write(primary user) + Deny Write(all other users).

Share
Opens files as Read/Write(primary user) + Deny None(all other users).

Other
Specify a custom combination of primary user + other user access.

User Access
Choose from Read Only, Write Only, or Read and Write.

Other Access
Choose from Deny None, Deny All, Deny Read, Deny Write, or Any Access (FCB compatibility mode).

Individual File Overrides

Select the **Individual File Overrides** tab to override data dictionary settings for individual files in the data dictionary. Highlight the file whose attributes you want to change, then press the **Properties** button.

The prompts on this tab mirror those on the File Control tab, and they behave exactly the same way, with these exceptions.

- The settings here apply only to the single file selected.
- Each drop-down list provides an additional choice: *Use Default. Use Default* sets the attribute according to the **File Control** tab.

File Manager Options for *Filename*
Press this button to display the File Manager Options dialog. You may derive your own new methods or properties for the File Manager Class, or you may specify your own third party class to override the default File Manager Class.

Relation Manager Options for *Filename*
Press this button to display the Relation Manager Options dialog. You may derive your own new methods or properties for the Relation Manager Class, or you may specify your own third party class to override the default Relation Manager Class.

File Attributes
These settings reflect the same settings found on the **File Control** tab, but apply only to the individual file currently selected.

File Declaration Mode
Select from the drop down list one of the four following options. The table in which you are overriding properties for will be generated as a FILE, QUEUE, or GROUP.

Use User Options
The table is generated based on the user options defined in the dictionary. The user options can be set to DATA,QUEUE or DATA,GROUP. This will cause the table to be generated as a QUEUE or GROUP respectively. No user options will cause the table to be generated as a FILE. The **File Access** prompts for **User Options** are the same as the prompts on the **File Control** tab.

As FILE
The table will be generated as a FILE structure. The File Access prompts for User Options are the same as the prompts on the File Control tab.

As GROUP
The table will be generated as a GROUP structure with the possibility of the attributes below.

THREAD
Check this box to add the THREAD attribute to the GROUP structure.

BINDABLE
Check this box to add the BINDABLE attribute to the GROUP structure.

NAME
Specifies the value of the NAME attribute added to the GROUP structure.

OVER
Specifies the value of the OVER attribute added to the GROUP structure.

TYPE
Check this box to add the TYPE attribute to the GROUP structure.

As QUEUE
The table will be generated as a QUEUE structure with the possibility of the attributes below.

THREAD
Check this box to add the THREAD attribute to the QUEUE structure.

BINDABLE
Check this box to add the BINDABLE attribute to the QUEUE structure.

NAME
Specifies the value of the NAME attribute added to the QUEUE structure.

OVER
Specifies the value of the OVER attribute added to the QUEUE structure.

File Access

File Open Mode
Specifies how your application shares files among concurrent users. See the *Language Reference* for more information.

Open Opens file as:
Read/Write (primary user) +
Deny Write (all other users).

Share Opens files as:
Read/Write (primary user) +
Deny None (all other users).

Other Specify a custom combination of primary user + other user access.

User Access
Choose from Read Only, Write Only, or Read and Write.

Other Access
Choose from *Deny None*, *Deny All*, *Deny Read*, *Deny Write*, or *Any Access* (FCB Compatibility mode).

Defer Opening File
Specifies when your application opens related files. Select Yes to delay opening the file until it is accessed. Delaying the open can improve performance when accessing only one of a series of related files. Select No to open the file immediately whenever a related file is opened. See *File Manager Class – Lazy Open* and *Use File* for more information.

External Module Options

Select the **External Module Options** tab to set options associated with your application's external modules. This tab is only available when your application contains an external module (LIB or DLL). Select the external module whose attributes you want to change, then press the **Properties** button.

Standard Clarion 6 LIB/DLL
Check this box if the LIB or DLL is produced by the ABC Templates or a similar coding scheme. Checking the box generates code to initialize and shut down global objects used by the LIB or DLL. If it is a hand-coded LIB or DLL you should probably clear this box.

Global Objects

This tab lets you specify the default object names the global objects used in an application. You can also specify the default classes to be used for the global objects.

Don't generate globals
Check this box to not generate any global data for the application. This includes FILEs and object declarations. It allows you to generate one DLL that is shared by all applications.

Error Manager
Press this button to display the Error Manager classes dialog. You may derive your own new methods or properties for the Error Manager Class, or you may specify your own third party class to override the default Error Manager Class.

INI File Manager
Press this button to display the INI File Manager classes dialog. You may derive your own new methods or properties for the INI File Manager Class, or you may specify your own third party class to override the default INI File Manager Class.

Run-time Translator
Press this button to display the Run-time Translator classes dialog. You may derive your own new methods or properties for the Run-time Translator Class, or you may specify your own third party class to override the default Run-time Translator Class. (Checking the Run-time translation check box on the General Tab enables this button.)

Fuzzy Matcher
Press this button to display the Fuzzy Matcher classes dialog. You may derive your own new methods or properties for the FuzzyClass, or you may specify your own third party class to override the default Fuzzy Matcher class. (Checking the Enable Fuzzy Matching check box on the General Tab enables this button.)

Classes

Enable the use of ABC Classes (Clarion Family Only)
Check this box to allow the ABC Classes to be used with the Clarion template family. This is required for certain templates that are used in both families that reference ABC Classes. The default is on (checked). The classes enabled by this check box are listed in the Default Classes group, and include all classes regarding edit-in-place, reports, and calendar support.

Refresh Application Builder Class Information
Press this button if you have changed the contents of or added an include file (.INC) to the \LIBSRC directory. Typically, this is needed when you install third party products that use ABC compliant classes, although you may create your own ABC compliant classes too. The ABC Templates use information gleaned from the header files for generating embed points, loading the Application Builder Class Viewer, application conversion, etc.

Application Builder Class Viewer
Press this button to display classes, properties, and methods used by the ABC Templates, and the relationships between parent and derived (child) classes. This utility can help you analyze and understand the classes that the ABC Templates use.

ABC Library FilesTPLApplication_ABC_Library

Task Grouping Buttons
Each task-grouping button identifies tasks or types of tasks the ABC Templates accomplish. Each button lets you specify the class or classes the ABC Templates use to accomplish the tasks named by the button's text. Following are the ABC Template tasks and their associated default classes.

General Tasks

This window allows you to specify the name of the Class that handles certain tasks in your application. You may specify alternate classes by typing the class name in the corresponding entry field. The class you name must be an ABC compliant class.

The Configure buttons allow you to set options that affect the behavior of the named class.

Configure WindowManager

Reset on gain focus
Check this box to make the WindowManager unconditionally reset whenever the window receives focus. Clear the box to allow a conditional reset (reset only if circumstances demand, for example, when the end user invokes a new BrowseBox sort order or invokes a BrowseBox locator).

Auto Tool Bar
Check this box to make the WindowManager try to set the appropriate ToolbarTarget whenever the end user selects a new TAB control. Clear the box to manually set the ToolbarTarget or use the current ToolbarTarget.

Auto Refresh

Check this box to make the WindowManager automatically reset the window and its associated objects whenever it detects a change. The WindowManager checks for changes after it processes each event. AutoRefresh is particularly useful when the resetting of a Browse Box changes a field which is a range-limit of another Browse Box.

Configure ErrorManager

Default Error Category
Specify a default error category for errors that do not have their own category. The default is ABC.

Allow Select & Copy of Message Text
Check this box to allow your users to highlight, copy and paste message text of posted error to the clipboard for relaying the information to the appropriate parties..

Store Error History
Check this box to save error history to be viewed at a later time during the session.

Limit Stored History
Check this box to limit the number of history items the history queue can hold at a single instance.

History Threshold Limit
Specifies the maximum number of items to hold in history. This is available if **Limit Stored History** is checked.

View Trigger level
Select an error severity level from the drop-down list. When the error level occurs the action is recorded to history. The available error levels are Level:Benign, Level:Cancel, Level:Notify, Level:User, Level:Program, and Level:Fatal.

Configure Resizer

Automatically find parent controls
Check this box to make each Resizer object set parent/child relationships among window controls. Clearing the box makes the WINDOW the parent of all its controls. Setting parent/child relationships lets any special scaling cascade from parent to child.

Optimize Moves
Check this box to move all controls at once during the resize operation, producing a snappier resize and avoiding bugs on some windows.

Optimize Redraws
Check this box to make controls transparent (TRN attribute) during the resize operation, producing a smoother redraw and avoiding bugs on some windows.

Configure Run-time Translator

Extract Filename
Specify a filename to receive a list of all runtime text that may require translation for multi-language applications.

This is only available if you have checked Enable Run-Time Translation in the Application's Global Properties.

Configure Calendar

Change Default Color
Check this box to allow you to ioverride the default colors of the selected CalendarClass. You can change the color for Sunday, Saturday, Holiday, and all Other dates.

Action for Close Button
Each Calendar Class has a default Select and a default Close button. If you would like both buttons to return the current date selected, choose *Select and Close* from the drop list. If you only want the Select button to return the date, choose *Cancel* from the drop list.

File Management Tasks

Browser Tasks

This window allows you to specify the names of the Classes which handles certain tasks in your application's browsers. You may specify alternate classes by typing the class name in the corresponding entry field, or select a class name in the drop list provided. The class you name must be an ABC compliant class.

The **Configure** button allows you to set options that affect the behavior of the Browser class. Additional browser related classes are configured by selecting the appropriate tab control at the bottom of this dialog window. The classes that you can configure at the time of this release are EIP (Edit-In-Place), QBE (Query-By Example), Step and Locator Managers, and Others (which include Fuzzy, Grid, and Sidebar Managers).

The EIP Configuration has a special Template Interface prompt which controls the level of configuration that you can apply to your target browse. Select *Original* to control EIP behavior with limited template prompt control. Select *Detailed* for more specific control for each column that has EIP enabled.

For more information concerning these classes, refer to the ABC Library Reference.

Configure Browser

Active Invisible
Check this box to fill the browse queue even when the browse LIST is "invisible" because it is on a non-selected TAB or is otherwise hidden. This improves performance for procedures with invisible browse lists; however, buffer contents for the invisible browse list should not be relied upon. Clear the box to suppress the refill when the listbox is hidden.

Allow Unfilled
Check this box to allow a partially filled LIST when the result set "ends" in mid-list. This improves (SQL) performance by suppressing additional reads needed to fill the list. Clear the box to always display a "full" list.

Retain Row
Check this box to maintain the highlight bar in the same list row following a change in sort order, an update, or other browse refresh action. Clear the box to allow the highlight bar to move.

Process & Report Tasks

This window allows you to specify the name of the Class which handles certain tasks in your application. You may specify alternate classes by typing the class name in the corresponding entry field. The class you name must be an ABC compliant class.

Ascii Viewer Tasks

This window allows you to specify the name of the Class which handles certain tasks in your application. You may specify alternate classes by typing the class name in the corresponding entry field. The class you name must be an ABC compliant class.

Toolbar Manager Tasks

This window allows you to specify the name of the Class which handles certain tasks in your application. You may specify alternate classes by typing the class name in the corresponding entry field. The class you name must be an ABC compliant class.

ABC Library Files
This dialog allows the ABC libraries to be linked in to the application, linked in as an external DLL, or linked in as an external library. If you choose external DLL or external library, the external library base name must be specified.

Clarion Version Information

This tab displays the latest version and family information for the active template set. This information is useful for your own documentation and SoftVelocity Technical Support for reporting purposes.

Global Classes Window

Many of the ABC Procedure, Control and Extension templates provide a Classes tab or dialog. These local Classes tabs let you control the classes (and objects) your procedure uses to accomplish the template's task—that is, they override the global class settings specified here in the **Global Properties** dialog. Deriving your own class can give you very fine control over the procedure when the standard Application Builder Class is not precisely what you need.

Object Name
Set the object's label for the template-generated code.

Use Default Application Builder Class
Check this box to use the default Application Builder Class. Clear this box to use a class other than the default, and to enable the following prompts.

Use Application Builder Class
Check this box to select a class from the **Base Class** drop-down list. The list includes all classes with the LINK attribute in \LIBSRC*.INC files. Clear this box to specify a class declared elsewhere.

Base Class
If you checked the **Use Application Builder Class?** box, select a class from the drop-down list. If you cleared the **Use Application Builder Class?** box, type the class label here, and type the name of the source file that contains the class declaration in the **Include File** entry box.

Include File
If you cleared the **Use Application Builder Class?** box, type the class label in the **Base Class** entry box, and type the name of the source file that contains the class declaration here.

Derive
Check this box to derive a class based on the parent class specified above and to enable the **New Class Methods** and **New Class Properties** buttons to define any *new* properties and methods for the derived class.

This prompt is primarily to allow you to define *new* properties and methods in a derived class. To override *existing* methods, simply embed code in the corresponding method embed points.

Using **Derive?**, **New Class Methods** and **New Class Properties** makes the template generate code similar to the following:

```
MyProcess CLASS(Process) !derive a class from the parent class
NewMethod   PROCEDURE    !prototype new class method
NewProperty BYTE         !declare new class property
          END
```

The template automatically derives from the parent class if you embed code into any of the derived method embed points, regardless of the status of this check box.

New Class Methods
Press this button to specify the *new* method prototypes to generate into the derived CLASS structure. This opens the **New Class Methods** dialog (see *New Class Methods*).

New Class Properties
Press this button to specify the new property declarations to generate into the derived CLASS structure. This opens the **New Class Properties** dialog (see *New Class Properties*).

Refresh Application Builder Class Information
Press this button if you have changed the contents of an include file (.INC) or added an include file to the \LIBSRC directory. Typically, this is needed when you install third party products that use ABC compliant classes, although you may create your own ABC compliant classes too. See *ABC Compliant Classes* for more information.The ABC Templates use information gleaned from the header files for generating embed points, loading the Application Builder Class Viewer, application conversion, etc.

Application Builder Class Viewer
Press this button to display classes, properties, and methods used by the ABC Templates, and the relationships between parent and derived (child) classes. This utility can help you analyze and understand the classes that the ABC Templates use.

Composite Class
Press these buttons to open a Classes dialog for each class used by the parent class specified above. For example, the WindowManager uses a Toolbar class, so the WindowManager's Classes dialog contains a Toolbar Class button to open a Classes dialog for its Toolbar Class.

List Format Manager Configuration

The List Format Manager is designed to give you the ultimate control of your Browse Box formatting (display options) to your users at runtime. You are essentially creating a table (file) that stores an unlimited amount of list box formatting strings, which are read and applied to your target browse box at runtime. Other stored formats are available via a popup menu, as well as a Format Editor to define new ones. These options are set on the procedure level, and can be accessed here.

Note:

When using the Listbox Format Manager during development, removal of a column from the list box may result in errors if the removed column is still referenced in a previous saved list box format. To avoid this, modify the saved listbox format prior to removing any referenced columns from the listbox.

The list format file can be stored in the active data dictionary, or defined within the specific application.

The following prompts are provided:

UserID Field
Enter a number to assign to all list formats used by this application. You can assign a different number to another application, which will only access special formats for that application.

Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the appropriate prompt.

Table Origin
Your table that stores the list format information can be created by your Application, or can be a table that is defined in your Dictionary.

We recommend that you allow the Format Manager to create a file for you (Application), and later import it into an existing dictionary. This will allow you to share formats between different applications that use the same data dictionary.

Table Configuration

If your runtime format file is to be defined in the application (instead of dictionary storage), press this button to access the Table Configuration dialog.

Dictionary Table
If you have selected Dictionary as the Table Origin choice, press the ellipsis to select the appropriate table name from your active dictionary.

The default file format is a TOPSPEED file, and is shown as follows:

```
LFM_CFile FILE,PRE(CFG),CREATE,DRIVER('TopSpeed'),THREAD,NAME('Formats.FDB')
key_Main  KEY(+CFG:ProcId,+CFG:UserId,+CFG:CtrlId,+CFG:FormatId),OPT,NOCASE
Record    RECORD,PRE()
AppName     STRING(30)                      ! Name of Application
ProcId      STRING(30)                      ! Procedure identifier
UserId      SHORT                           ! User identifier
CtrlId      SHORT                           ! Control identifier
FormatId    SHORT                           ! Format identifier
FormatName  STRING(30)                      ! Format name
Flag        BYTE                            ! Default/current flag
Format      STRING(5120)                    ! Format buffer
VarLine     STRING(2048)                    ! Variable buffer
          END
         END
```

The file driver that you use from the dictionary may be modified, provided that the field and key labels are not modified.

Some elements of the format file can be modified as shown in the Table Information dialog.

SortOrder's menu text

Enter alternate text (or a variable name using *!variable* format) that the List Format Manager will use to store the menu text used in generated "Sort Order" popup item. This allows for translation to alternate languages (e.g., Spanish) if needed.

Module Properties

This dialog lets you specify settings for an individual source code document file, i.e. a module. To access this dialog select the **Module** tab, highlight a module (folder icon), then press the **Properties** button.

Name	Specifies the file name for the module.
Description	A short description which appears in the Application Tree when in Module View.
Type	Press the ellipsis (...) button to choose from the Select Module Type dialog.
Map Include File	Specify a source code file to include in the MAP section of the module.
Allow Repopulate	Check this box to let the Application Generator move procedures from this and other modules.

Module Data is now located on the Data / Tables Pad

Embeds Tab
Calls the Embedded Source dialog for module level embed points--data declarations.

See Also: Adding Embedded Source Code

New Procedure Dialog

This dialog allows you to add a new procedure to the Application Tree. Generally, you use this command to add a source code procedure, which you can then call from Embed Points accessed from other procedures.

Type a new procedure name in the dialog, then choose from the **Select Procedure Type** dialog.

Picture Editor (Edit Picture)

Some controls let you specify a picture token that provides a special format for display or printing of variables

There is a great variety and diversity of picture token syntax that depends on the type of data you format: strings, numbers, currency, scientific, dates, times, etc.

The **Edit Picture** dialog lets you quickly and easily build an appropriate picture token without memorizing picture token syntax. Invoke this easy to use dialog by pressing the ellipsis (...) button beside the **Picture** prompt in the properties dialog.

Example An example of the display format currently specified in the dialog. What you see is what you get.

Pool Select a predefined picture from the drop list.

Save As Press this button to save the displayed picture to the Pool, and name the saved picture. The saved pictures are available in the **Pool** drop-list. You can save your most frequently used pictures, and then quickly reuse them from the Pool. **Note: The Picture Editor stores the pictures in the ..\BIN\C70Pict.ini file.**

Delete Press this button to delete the displayed Pool picture currently displayed in the **Pool** field.

Picture The picture token currently specified. This picture token produces the example shown.

Picture Type Select the type of data to format from this drop-down list. Choose from:

StringGHZKLW

Numeric and Currency

Scientific Notation

DateJJ3BLP

TimeKTQ71X

PatternBO0ZOK

Key-in Template

String

String Pictures specify a length with no other formatting.

Length
Specify the length of the string. This length also determines the width of the control if the width is not otherwise specified by the control's AT attribute.

Numeric and Currency

Numeric and Currency picture tokens specify a length, plus conventional formatting to convey positive and negative values, various currencies, etc.

Size
The total number of significant digits, plus any formatting characters. For example, $22.25- is 4 significant digits + 3 formatting characters for a size of 7.

Decimal Digits
The number of digits to the right of the decimal.

Currency
Choose from None, Leading, and Trailing. None shows no currency symbol. Leading puts the currency symbol to the left of the number and Trailing puts the symbol right of the number.

Symbol
The currency symbol to display: either a dollar sign ($) or a string constant.

Negative Sign
Specify how negative values are formatted.

Bracket
Negatives surrounded by parentheses.

Leading
Negatives get a leading minus sign.

Trailing
Negatives get a trailing minus sign.

None
No sign display.

Decimal Separator
Specify the character inserted between the integer and fractional portion of the value.

Period
Period is the separator.

Comma
Comma is the separator.

None
Displays no separator.

Grouping
Specify the character inserted at every third digit to aid readability.

Comma
Comma is the separator.

Period
Period is the separator.

Space
Space is the separator.

Hyphen
Hyphen is the separator.

Leading Character
Specify the character to represent leading zeroes.

Clip
Remove leading zeroes so that any leading format characters abut the left most digit.

Zero
Leading zeroes display as zeroes (0).

Space
Leading zeroes display as spaces ( ).

Asterisk
Leading zeroes display as asterisks (*).

Blank When Zero
Check this box to display nothing when the value is zero.

Scientific

Scientific Notation picture tokens let you display very large or very small numbers with a decimal format raised by a power of ten. The display takes the form -9.99e+999.

Number of Characters	The total number of characters, including the 7 format characters. For example, -1.96e+007 requires 10 characters.
Leading Digits	The number of digits to the left of the decimal point (typically 1).
Decimal Separator	Specify the character inserted at every third digit to aid readability.
Point	Period is the separator.
Comma	Comma is the separator.
Space	Space is the separator.
Separator	Specify the character inserted between the integer and fractional portion of the value.
Point	Period is the separator.
Comma	Comma is the separator.
Blank When Zero	Check this box to display nothing when the value is zero.

Date

Date picture tokens let you display dates in a number of different formats. Choose the format you want from the **Format** drop-down list.

Better still, date picture tokens in entry fields automatically invoke Clarion's run-time date parsing functions, so you can enter '21' and Clarion expands it to the 21st day of the current month and year. Or you can enter 'DEC' and Clarion expands it to the 1st day of December of the current year. The date is then formatted according to the picture token.

The MASK attribute (Entry Patterns check box) on a window preempts the date parsing functions.

Format
Choose the format you want from the drop-down list. What you see is what you get except for the Windows Short and Windows Long formats. Additionally, the separator character and leading zeroes may be specified independent of the chosen format.

Windows Short
Uses the short date format specified in the Windows control panel or the Windows 95 Regional Settings control panel.

Windows Long
Uses the long date format specified in the Windows control panel or the Windows 95 Regional Settings control panel.

Separator
Choose from Standard (/), Period (.), Dash (-), Space ( ), and Comma (,).

Leading Characters
Specify the character to represent leading zeroes.

Zero
Leading zeroes display as zeroes (0).

Blank
Remove leading zeroes.

Asterisk
Leading zeroes display as asterisks (*).

Two digit date range
Change the default century interpretation for dates input with a two digit year. By default, Clarion assumes any date input with a two digit year (i.e. the century value is omitted) falls between today-**80** years and today+19 years.

For example, if today is June 1, 1996 and the date input is 9/2/59, Clarion assumes the 59 means 1959 since 1959 falls between today-80 years (June 1, 1916) and today+19 years (June 1, 2015). To force a different interpretation, set the **Two Digit Date Range** to 30. Now Clarion assumes the 59 means 2059 since 2059 falls between today-30 years (June 1, 1966) and today+69 years (June 1, 2065).

Blank When Zero
Check this box to display nothing when the value is zero.

Time

Time picture tokens let you display times in a number of formats. Choose the format you want from the **Format** drop-down list.

Format
Choose the format you want from the drop-down list. What you see is what you get except for the Windows Short and Windows Long formats. Additionally, the separator character and leading zeroes may be specified independent of the chosen format.

Windows Short
Uses the time format specified in the Windows control panel.

Windows Long
Uses the time format specified in the Windows control panel.

Separator
Choose from Standard (:), Period (.), Dash (-), Space ( ), and Comma (,).

Leading Character
Specify the character to represent leading zeroes.

Zero
Leading zeroes display as zeroes (0).

Blank
Remove leading zeroes.

Blank When Zero
Check this box to display nothing when the value is zero.

Pattern

Pattern pictures let you build custom display formats for various numbers: phone numbers, social security numbers, room numbers, dates, times, measurements, etc.

Picture
Type the picture token between the 'P's according to the Legend below. Your picture token can include any displayable characters, including all the standard keyboard characters.

At runtime, the constants in the picture token display just as they appear in the token. The left angle (<) and the pound sign (#) resolve into the individual digits from the display variable.

Legend

<	integer, blank if zero
#	Integer
constant	(any displayable char except < and #)

Blank When Zero
Check this box to display nothing when the value is zero.

To use a lowercase p in your picture, use an uppercase P at the start and end of your picture token. To use an uppercase P in your picture, use a lowercase p at the start and end of your picture token.

Key-in Template

Key-in Pictures let you build custom edit formats for runtime fields containing mixed alphanumeric characters. Although Key-in tokens affect output as well as input, their primary purpose is to provide custom field editing and validation on input.

Picture Type the picture token between the 'K's according to the Legend below. Your picture token can include any characters (displayable or not), including all the standard keyboard characters.

Legend

<	accept an integer, blank if zero
#	accept an integer
?	accept any character (even non-display)
^	accept an upper case character
_	accept a lower case character
|	input may stop here
constant	(any displayable char except <#?^_| or \)

\ display next char (lets you display <#?^_| or \)

Only Alphabetic Characters
Check this box to accept only alphabetic characters.

Blank When Zero
Check this box to display nothing when the value is zero.

To use a lowercase k in your picture, use an uppercase K at the start and end of your picture token. To use an uppercase K in your picture, use a lowercase k at the start and end of your picture token.

Program Properties

This dialog lets you specify global settings for the application's main program. To access this dialog select the **Module** Tree Mode, highlight the first item (the program), then press the **Properties** button.

Name	Specifies the name of the program's source file.
Description	A short description which appears in the Application Tree when in Module View.
Type	Press the ellipsis (...) button to choose from the **Select Program Type** dialog.
Map Include File	Specify a source code file to include in the MAP section of the module.
Allow Repopulate	Check this box to let the Application Generator move procedures from this and other modules.

Global Data elements are located in the Data / Tables Pad.

Embeds
Calls the **Embedded Source** dialog for global level embed points including the global map, global data, file open routines, the export file, and the ship list.

See Also: Adding Embedded Source Code

Set Visual Indicators

This dialog window allows you to specify the properties to apply to targeted controls when they are selected. Choose from the following options:

Include These Controls

Check the appropriate box to allow the Visual Indicator properties to be applied to specific control types. Choose from TEXT, SPIN, RADIO, CHECK, DROP(List and Combo), and LIST control types. When visual indicators are active, ENTRY controls are always enabled.

Control Color Indicator Tab

Change Color
Check this box to allow a control to change its color and text properties when it receives input focus (e.g., is selected).

Background Color
Press the ellipsis button to select the color to use as the background of a control when it is selected.

Bold Text
Check this box to allow any text displayed and entered to be shown in Bold text. When you move to the next control, the text returns to normal.

Exclude DropList
Check this box if you do not wish to apply the selected color entered above to your Drop List controls, but still wish the Bold Text option to be applied. If you want to disable the Visual Indicator feature entirely for all Drop List controls, uncheck the appropriate box located in the **Include These Controls** section above.

Required fields - Override settings
Check this box if you would like any fields with the REQ attribute set to be handled with a different color than other selected fields.

Required Fields – Background Color
Press the ellipsis button to select the color to use as the background of the selected required entry or text control.

Box Indicator Tab Control

Display Box
Check this box to allow a box (with color options) to appear around any control with selected focus. When enabled, the color, border, and size options that follow are enabled.

Fill Color
Press the ellipsis button to select the color to use as the box fill used with the selected control.

Border Color
Press the ellipsis button to select the color to use as the box border used with the selected control.

Size
Set the thickness of the box's border by entering a point value in the spin control. The default is 2 points.

Override settings
Check this box if you would like any fields with the REQ attribute set to be handled with a different color than other selected fields.

Required Fill Color
Press the ellipsis button to select the color to use as the box fill used with the selected control that is required.

Required Border Color
Press the ellipsis button to select the color to use as the box border used with the selected control that is required.

Visual Indicator Tab Control

Display Indicator
Selected fields can also have a special indicator that will be placed immediately to the left side of the selected control type. Check this box to activate a visual indicator.

Indicator Color
Press the ellipsis button to select the color to use with the visual indicator.

Indicator Character
Enter a special character string that you wish to use as the visual indicator
(Example: >>)

Offset
Enter the distance in dialog units that separate the visual indicator from the actual selected control.

There is also an **About** tab control located on this dialog window that provides additional information about this feature.

Table Configuration

The List Format Manager's Table Configuration dialog is used to define your table (file) parameters that will be used to create the list box format configuration file. This file is valid only for this application, but can later be imported and stored in your data dictionary. Note: If you wish to create a runtime format file that is used in multiple applications, define and reference it through a data dictionary.

The following prompts are presented to help you define your format file:

File Group:

Name
Enter a valid file label name to use as the format file. The default name is *LFM_CFile*

Prefix
Enter a prefix to use for the format file. The default prefix is *CFG*

File Attributes Group:

Check the **OEM**,**Thread**, and **Encrypt** boxes if you wish to apply these attributes to your format file. We recommend that you check the Thread box, to allow the List Format to be used across different thread instances.

Password
Enter an optional password if you wish to encrypt this file for any reason.

Record Group:
The record group allows you to tailor the format file size. For example, if you are using very large list boxes, you may need to increase the format size.

Name Size refers to the size of the Format name that will display in the popup menu and Format Editor. *Format Size* holds the list box FORMAT string, and Variable Size holds your formatting options. All all STRING data types, and should be sized accordingly.

File Name Group:

NameThe **Default** name of the format configuration file is *FORMATS.FDB*. If you need to change this for any reason, choose *Other* from the drop down list, and enter the new name in the **Use Name** prompt provided. When *Other* is selected, you may also modify the path of the format configuration file location. If the name of the format configuration file is stored in a variable, select the *Variable* option, and enter the variable name (*without* the prepending exclamation point) into the **Variable Name** prompt provided.

PathThe default path of the format file is the program's directory (folder). If you need to store this file in another location, enter the appropriate path.

Make sure that your applications that share a common directory specify a different name or path for each application. This can avoid possible conflicts with stored format strings.

Table Relationship Dialog

Dictionary
Select this option to use the relationship as defined in the Dictionary.

Custom
Select this option to create an ad-hoc join between the files. This generates a JOIN statement.

Inner
Check this box to specify that this is an Inner join. This adds the INNER attribute to the JOIN statement.

XP/Vista Manifest Support

Include Default XP Manifest
Check this box to include a default Manifest file in your application. When your program runs in the XP environment, this instructs the XP theme manager to render your controls with the new look and feel of the Windows XP environment. The manifest file is ignored on other Windows platforms (95/98, 2000, NT, and Vista). This option add the necessary MANIFEST logic to the application's export file.

The above setting should be used for XP Manifest only (e.g., applications running on a Windows XP O/S). If you plan to use a *Vista* Manifest, use the following options instead:

Generate Manifest
Check this box if you need to export an external XP manifest file that is associated with this application. This external manifest file is required when migrating your application to Clarion 7, to support the latest XP visual styles and the extended UI support.

Link Generated Manifest in Project
Check this box to automatically include the external Manifest generated above into your distributed applications. This option eliminates the need for you to include the *.manifest* file with your application.

Configure Manifest for Vista
Check this box to enable additional options to be used with the Vista manifest. These options are explained in detail below:

If you attempt to run an application on UAC-enabled Vista without an embedded manifest file, any code or user action event attempting to display a simple file dialog can throw unhandled security exceptions. There are many scenarios where you might need to elevate the UAC access level for standard users, especially when porting a legacy application to Vista, and a complete UAC-refactor is not possible. There are three levels available in an application manifest for invoking requested execution privileges.

Vista Execution Level

Choose from one of three execution privilege levels:

asInvoker
The standard user token is used to start the process.

If your application doesn't require any special privileges, use the *asInvoker* execution level. Your users will run the program transparently without encountering a Vista popup each time they invoke it. Attempted writes to forbidden areas will fail rather than being virtualized.

highestAvailable
The application runs with the highest privileges the current user can obtain.

If your application doesn't require a full Administrator account but does require certain privileges, such as those accorded to accounts that are members of the Backup Operators group, use the *highestAvailable* execution level. An ordinary user will not encounter the elevation prompt and your app will not succeed in exercising the elevated privileges. A user who has one or more of the privileges that trigger Vista elevation will encounter the Vista elevation popup when he runs your application. If his account has sufficient privileges, your application will function as intended; otherwise not.

requireAdministrator
Prompt standard users for administrative credentials.

If your application requires an account that is a member of the Administrators group, use the *requireAdministrator* execution level. This is necessary if your application writes to certain areas of the Windows Registry such as HKEY_LOCAL_MACHINE, needs to write within C:\Program Files or C:\Windows, needs to install drivers, or needs to perform other tasks that ordinary users are restricted from doing. Each time your application is run, the user will encounter the Vista elevation popup and will need to authorize your app to run.

In summary, the execution level you choose for your manifest will depend on the requirements of your application.

Note:

Applications ported from early versions of Windows that could (if built for Vista) work as standard-user applications may require administrative token access via the credentials prompt or administrative approval mode until they can successfully incorporate the best UAC practices.

Application requires UIAccess

Check this box to set the UIAccess to "True". In this state, the application is allowed to bypass UI protection levels to drive input to higher privilege windows on the desktop. This setting should only be used for UI Accessibility applications.

With this option unchecked, the target application does not need to drive input to the UI of another window on the desktop. Applications that are not providing accessibility should set this flag to false. Applications that are required to drive input to other windows on the desktop (on-screen keyboard, for example) should set this value to true.

Do you really need UIAccess?

This capability is generally intended only for accessibility utilities. If you do need UIAccess enabled, then the executable needs to be digitally signed, and must be installed under %windir% or %ProgramFiles%.

If **Application requires UIAccess** is set to FALSE (unchecked):
The application does not need to drive input to the UI of another window on the desktop. Applications that are not providing accessibility should set this flag to false. Applications that are required to drive input to other windows on the desktop (on-screen keyboard, for example) should set this value to TRUE.

If **Application requires UIAccess** is set to TRUE (checked):
The application is allowed to bypass the UI protection levels to drive the input to higher privilege windows on the desktop. This setting should only be used for UI Accessibility applications.

UIAccess is only enabled if the Vista execution level is set to *requireAdministrator*.

Make controls inside the TAB transparent
Sets the transparent (TRN) attribute to every OPTION, GROUP, RADIO, STRING, CHECK, and PROMPT controls that are placed inside a TAB control. This gives these controls a more pleasing appearance when displayed inside a TAB control that uses the new Visual Styles.

Tree Mode Views

Application Tree Dialog - Category View

In this view, the **Application Tree** dialog displays your procedures grouped by the Category you specify on the Procedure Properties dialog. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.

Application Tree Dialog - Modified View

In this view, the **Application Tree** dialog displays your procedures sorted by the last modified date and time. The most recent changed procedure appears at the top of the list, while the oldest procedure appears at the bottom. If you ever need to trace when a problem arose, this view can assist you in which procedure could be the problem.

Application Tree Dialog - Module View

In module view, the **Application Tree** dialog displays your procedures according to the source code document which they reside in. A procedure is a collection of instructions--Clarion language statements--which perform a task. The first procedure your application executes is called "Main" by default.

The tree controls in this dialog illustrate how the procedures reside in each separate file. Within each, they branch from each other when parent and child procedures reside in the same file.

When you right-click on any procedure and select **Module** from the pop up menu, the Editor stores certain information about the files you open to edit. This is a productivity aid; it remembers the line number you were last at, and so if a file is opened, positioned to a certain line number, closed, then reopened, you are now back at the line where you were last positioned.

Application Tree Dialog - Name View

In alphabetic view, the **Application Tree** dialog displays your procedures according to the order of the procedure names. A procedure is a collection of instructions--Clarion language statements--which perform a task. The first procedure your application executes is called "Main" by default.

Alphabetic view makes it easier to locate the precise procedure you wish to edit, when your application includes many procedures. There is also a **Find** command on the **Edit** menu, to help you locate the procedure you want when the Procedure tree is *very* long.

The Application Tree shows the procedures you create when you add a menu item, toolbar command, or an embedded source procedure. Each new procedure is marked "To Do." When you "fill in" its functionality, the Application Tree dialog replaces the "To Do" with your description.

Application Tree Dialog - Procedure View

In this, its default view, the **Application Tree** dialog displays your procedures in logical call tree, nesting each procedure under its calling procedure. A procedure is a collection of instructions--Clarion language statements--which perform a task. The first procedure your application executes is called "Main" by default.

The tree controls in this dialog illustrate how the procedures branch from "Main" and from each other. This provides a schematic diagram of your program's logical structure.

The Application Tree shows the procedures you create when you add a menu item, toolbar command, or an embedded source procedure. Each new procedure is marked "To Do." When you "fill in" its functionality, the Application Tree dialog replaces the "To Do" with your description.

Application Tree Dialog - Template Type View

In template type view, the **Application Tree** dialog groups all your procedures according to template type. A procedure is a collection of instructions--Clarion language statements--which perform a task. The first procedure your application executes is called "Main" by default.

The Application Tree shows the procedures you create when you add a menu item, toolbar command, or an embedded source procedure. Each new procedure is marked "To Do." When you "fill in" its functionality, the Application Tree dialog replaces the "To Do" with your description.

Procedure Properties - Called Procedures

Procedures may call other procedures. Procedure calls specified with template prompts are automatically added to the Application Tree; however, the Application Generator cannot "see" procedures called from embedded source code.

Identifying these embedded procedure calls is important to project management and to source code generation. When you use the **Procedures** button to identify procedures called within embedded source code, the Application Generator can properly display the procedures within the Application Tree hierarchy, *and* the Application Generator can generate the appropriate local MAP structure for the source module.

In addition, you can also use this feature for documenting external procedure calls and tables referenced therein.

The **Called Procedure** dialog provides two ways to add procedures:

Selection This tab control provides a complete list of all procedures defined in the application. Click on any procedure to identify it as a called procedure. You can also use the Select All and Deselect All to mark and unmark all procedures in the list.

List This tab control provides an alternative method for adding called procedures. This technique is useful when there are too many procedures in your application and the Selection method becomes difficult. In the List section, you manually enter, change, delete and arrange the called procedures using the buttons provided.

Procedure Properties

These dialogs--each is customized according to the procedure template--contain entry boxes in which you can add a text description for the procedure, or specify its source code module, plus command buttons which lead to the dialogs which allow you to customize the procedure.

Each procedure has its own custom help page which you can access by pressing the **Help** button on the **Procedure Properties** dialog. If you are using a third party template, or a template which you wrote yourself, this help topic will appear.

Therefore, this help page only describes the essential elements of the **Procedure Properties** dialog; the controls which each procedure template builds upon.

Clarion's Template language allows the template writer to add controls to the **Procedure Properties** dialog. These controls vary from template to template. Since each template performs a different task, the template writer provides whatever controls and options are necessary to gather input from you, the developer. Most of your input is stored in template variables (Template Symbols). When generating code, the Application Generator processes the template language code, and fills in the Template Symbols with the options you specify. As it does so, it generates your application's source code.

A **Procedure Properties** dialog could have, for example, a checkbox to specify that an MDI window should save its position in the .INI file between sessions. Each template adds controls such as these to the **Procedure Properties** dialog, to gather choices from you. At code generation time, the Application Generator evaluates a symbol which stored your choice as to whether you wanted to save the MDI window position. If the checkbox was marked "yes," the Application Generator processes the template code containing the executable code to support saving the window position, and writes it to the generated source code file.

Procedure Name	The **Procedure Name** is displayed here. Press the ellipsis button to change the name if desired. You will be prompted for a New Procedure Name, and the ability to Accept or Discard your changes.
Template	The base template type used by this procedure is displayed here. You can change the template type by pressing the ellipsis button to the right. The *Select Procedure Type* dialog is displayed for a new selection if needed.
Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	A category is used to help you group procedures together when the **Category** Tree Mode is active.
Module Name	The source code file to hold the code for the procedure. Select from the dropdown list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Prototype	Allows you to optionally type a custom procedure prototype which the Application Generator places in the MAP section.

Actions — If any control templates were pre-defined in the current procedure template, or were in a window or report by you, this button accesses the **Actions** dialog for the control templates.

Control templates provide "off the rack" controls, such as list boxes, *and* the code to maintain them. This allows you to start with a "bare" procedure template, such as the generic window, and add controls to create your own browse or form windows.

Parameters — Located in the *Procedure Properties Actions* dialog, this allows you to specify parameter names (an optional list of variables separated by commas) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code.

Return Value — Located in the *Procedure Properties Actions* dialog, this option lets you specify the variables receiving return values from functions (functions return values, procedures do not).

Procedure Tabs

In addition to the standard procedure prompts, there are **Procedure Tabs** that access other important procedure components:

Window — Calls the Window Editor and Designer, to visually design the window.

In the Window Editor, the **Edit as Text** button allows you to quick edit the WINDOW or APPLICATION structure at the source code level. Clarion allows you to easily switch back and forth between editing the window graphically, and editing the source code that describes it.

Press the **Full Editor** button to load and edit the WINDOW or APPLICATION structure in the full Text Editor. This editor enables full search capabilities, code folding, etc.

Take care when hand-editing code for any WINDOW which contains a control template. The Application Generator stores Template Language attributes which cannot be edited by hand (#ORIG, #LINK, etc.).

Report — Calls the Report Editor and Designer, to visually design the procedure report structure if applicable..

In the Report Editor, the **Edit as Text** button allows you to edit the REPORT structure at the source code level. Clarion allows you to easily switch back and forth between editing the report graphically, and editing the source code that describes it.

Press the **Full Editor** button to load and edit the WINDOW or APPLICATION structure in the full Text Editor. This editor enables full search capabilities, code folding, etc.

Calls References procedure calls made in hand-coded, embedded source.

Select this tab to access the Called Procedures dialog. To add a procedure, enter the procedure name in the **Add a New Procedure** section and press the A**dd** button.

If procedure calls already exists, the procedures appear selected in the **Called Procedures** dialog. To add another, simply select it. To delete a call simply deselect it. Additional buttons allow you to select all or deselect all as needed.

The purpose of the Calls tab is to add procedures called in embedded source code. The normal way to add template procedures to the Application Tree is to create a menu or toolbar command, add the procedure name via its Actions button, and let the Application Generator automatically add it to the tree.

Embeds Displays the **Embedded Source** dialog. You can then select either a field specific event or window related action, then add executable source code to customize how the procedure will handle it.

After you choose an embed point in the **Embedded Source** dialog, you choose the code to execute. You can specify a call procedure, which is then added to the tree. You can write your own code with the text editor. Or, you can choose and customize a code template, which is a combination of pre-written code and prompts to "fill-in-the-blanks."

Embedded source gives you complete control over *all* the processing in your procedures. It's one of the most powerful tools Clarion provides you.

Extensions Accesses extension templates, if any are installed on your system. Extension Templates allow additional functionality through "add-ins" to the Application Generator.

Formulas Accesses the **Formula Editor**, which allows you to create computed and/or conditional fields, which you can then reference in the controls you place in your windows and reports.

Procedure Properties--External

The External Procedure Template declares a procedure which is contained in an external library (*.LIB only) or object file. The Application Generator writes no source code. The project system links in the external file as a module.

After selecting the External template type from the **Select Procedure Type** dialog, choose OBJ or LIB from the **Select Module Type** dialog.

Type the file name of the external library or object file in the **Module Name** field. Optionally type parameter declarations in the **Prototype** field.

Description A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog.

Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.

Category Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.

Prototype Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator

Module Name The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module. **See Also: See Also:** Using DLLs not created in Clarion

The MODULE name for an External procedure should *not* be modified.

Declare Globally Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.

Export Procedure Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).

Tables This button is not valid for this procedure type.

Window The Window button is disabled for this procedure type.

Report The Report button is disabled for this procedure type.

Data This button is not valid for this procedure type.

Procedures Opens the **Called Procedures** dialog to add (or remove) a procedure to the Application Tree. To add a procedure, press the **Insert** button, then type the procedure name in the next dialog.

The purpose of the Procedure button is to add procedures called in embedded source code. In all other cases the Application Generator automatically adds procedures to the tree.

Embeds This button is not valid for this procedure type.

Formulas This button is not valid for this procedure type.

Extensions This button is not valid for this procedure type.

Procedure Properties--Frame

This template provides an MDI (Multiple Document Interface) parent frame, containing a predefined shell menu. The menu provides useful items such as an Exit command, plus the standard editing and window management commands.

When creating an MDI application, the Frame should be the main procedure. Use the Initiate Thread code template to start new execution threads for each MDI child window which you want to appear inside the frame.

Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.
Prototype	Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator
Module Name	The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Declare Globally	Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.
Export Procedure	Declares the procedure in the export file, enabling it to be called by another application. Note: This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).
Parameters	Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .
Return Value	For functions, lets you specify the variable receiving the return value.
Window Behavior	Press this button to control the behavior of the Window. See Window Behavior
ListBox Styles	Press this button to maintain any List Box Styles used by this procedure. See List Box Styles.
BIND Fields and	Press this button to maintain all BINDed variables and procedures used in

Procedures this procedure. See BIND Fields and Procedures.

Splash Procedure Names a procedure to call after the application frame opens, but before any user events are generated. Select from the drop-down list, or type a new procedure name.

By convention, a splash procedure provides a visual or audio (or both) fanfare for your program.

In addition to the standard procedure prompts, there are **Procedure Tabs** that access other important procedure components:

Window Calls the Window Designer, to visually design the window. **See Also:** How to Customize Your Window

The **Edit as Text** button next to the **Window** button lets you edit the WINDOW or APPLICATION structure at the source code level. Clarion lets you easily switch back and forth between editing the window graphically, and editing the source code that describes it.

Press the **Full Editor** button to load and edit the WINDOW or APPLICATION structure in the full Text Editor. This editor enables full search capabilities, code folding, etc.

Take care when hand-editing code for any WINDOW which contains a control template. The Application Generator stores Template Language attributes which cannot be edited by hand (e.g., #ORIG and #LINK).

Report Calls the Report Editor and Designer, to visually design the procedure report structure if applicable..

In the Report Editor, the **Edit as Text** button allows you to edit the REPORT structure at the source code level. Clarion allows you to easily switch back and forth between editing the report graphically, and editing the source code that describes it.

Press the **Full Editor** button to load and edit the WINDOW or APPLICATION structure in the full Text Editor. This editor enables full search capabilities, code folding, etc.

Calls References procedure calls made in hand-coded, embedded source.

Select this tab to access the Called Procedures dialog. To add a procedure, enter the procedure name in the **Add a New Procedure** section and press the A**dd** button.

If procedure calls already exists, the procedures appear selected in the **Called Procedures** dialog. To add another, simply select it. To delete a call simply deselect it. Additional buttons allow you to select all or deselect all as needed.

The purpose of the Calls tab is to add procedures called in embedded source code. The normal way to add template procedures to the Application Tree is to create a menu or toolbar command, add the procedure name via its Actions button, and let the Application Generator automatically add it to the tree.

Embeds

Displays the **Embedded Source** dialog. You can then select either a field specific event or window related action, then add executable source code to customize how the procedure will handle it.

After you choose an embed point in the **Embedded Source** dialog, you choose the code to execute. You can specify a call procedure, which is then added to the tree. You can write your own code with the text editor. Or, you can choose and customize a code template, which is a combination of pre-written code and prompts to "fill-in-the-blanks."

Embedded source gives you complete control over *all* the processing in your procedures. It's one of the most powerful tools Clarion provides you.

Extensions

Accesses extension templates, if any are installed on your system. Extension Templates allow additional functionality through "add-ins" to the Application Generator.

Formulas

Accesses the **Formula Editor**, which allows you to create computed and/or conditional fields, which you can then reference in the controls you place in your windows and reports.

Procedure Properties--Process

The Process procedure template sequentially processes a data file. You can specify a filter or range of on which records to perform the operation.. A predefined window contains a progress indicator to show the end user what percentage of the operation is complete.

Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.
Prototype	Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator
Module Name	The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Declare Globally	Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.
Export Procedure	Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).
Parameters	Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .
Return Value	For functions, lets you specify the variable receiving the return value.
Process Behavior	Press this button to control the behavior of the Window. See Window Behavior
ListBox Styles	Press this button to maintain any List Box Styles used by this procedure. See List Box Styles.
BIND Fields and	Press this button to maintain all BINDed variables and procedures used in

Procedures	this procedure. See BIND Fields and Procedures.
Process Properties Button	Press this button to define the process properties set up by the templates See below.

Process Properties

General

Window Message	The message displayed in the Progress Window dialog located just above the progress control, by default.
Action for Process	The action to perform for each record processed. **No record action** specifies no action to be performed by the process template. Use embedded source to handle the action. **PUT record** specifies that a record will be added. **DELETE record** specifies that each record processed will be deleted.
Use RI constraints on action	Check this box to enforce the RI constraints defined in your data dictionary. Clear this box to generate a simple PUT or DELETE depending on the **Action for Process** chosen.
Use Process Action to set Window Text	Check this box to place the **Action for Process** selection text in the Progress Window's title bar at runtime. If unchecked, the title text used in the Window Formatter will be the default.
Query Each Deletion	Check this box to specify that the user is prompted to confirm each record deletion.
MDI Progress Window	Check this box to specify that the Progress Window generated for this process will be MDI. This option is necessary if you plan to allow multiple processes to run on different threads at the same time. The disadvantage to this could be a slight reduction in process generation performance. DO NOT check this box if your process procedure is *not* threaded.
Progress Interval Timer	Enter a value in hundredths of a second that specifies the timer interval for this process. The lower the number, the progress update will show a smoother display during process generation. The larger the interval, the less frequent the progress window will be refreshed. Use this setting when you plan to run multiple processes at the same time, and need to fine tune the intervals between process procedures.
Quick-Scan Records	Specifies buffered access behavior for ODBC, ASCII, DOS, or BASIC files. These file drivers read a buffer at a time (not a record), allowing for fast access. In a multi-user environment these buffers are not 100% trustworthy for subsequent access, because another user may change the file between accesses. As a safeguard, the driver rereads the buffers before each record access. To *disable* the reread, enable QUICKSCAN.

Record Filter Type an expression to limit the contents of the process to only those records which match the filter expression. This filters all displayable records. When a Record filter is used in conjunction with a Range Limit, only those records within the specified range are filtered. **See Also:** Using Range Limits and Filters

You must BIND columns used in a filter expression. See Hot Fields below.

Additional Sort Fields Type a comma delimited list of fields on which to sort. These sort fields are in addition to the key for the report set in the **Data / Tables Pad** dialog. If no key is specified, this is the sort order used allowing you to sort records without a key.

Record Count Override When processing in record order (no key), this number is used to calculate what percentage of the operation is complete to provide feedback to the end user. If you don't specify a number, the process "counts" the records before processing begins. This can be relatively fast or slow depending on the file system and the file size. You must supply an appropriate record count when you use a Record Filter (or a Range Limit that results in a filter).

Set progress bar limits manually Clear this box to make your procedure read the result set and set the progress bar limits automatically. Setting limits automatically may produce poor performance for some SQL data sets, or erratic or inaccurate progress indicator for unevenly distributed result sets. Check this box to manually provide progress bar limits for the procedure. Setting manual limits can provide faster performance for SQL drivers and more accurate progress indicators for unevenly distributed result sets. This setting is only effective if you specify a Key for the File in the **Data / Tables Pad** dialog.

Low Progress Bar Limit Supply the lowest "free" key element value for the result set. You may type the value or the label of a variable containing the value. Enclose literal string values in single quotes ('value').

High Progress Bar Limit Supply the highest "free" key element value for the result set. You may type the value or the label of a variable containing the value. Enclose literal string values in single quotes ('value').

Range Limits

Range Limit Field Type in the field name or press the ellipsis (...) button to select the field from the Component list. The Range Limit Field must be a component of the Access Key specified in the Data / Tables Pad. The range limit is key-dependent; the generated source code uses the SET statement to find the first valid record.

Range Limit Type When a field is selected for **Range Limit Field**, this specifies the method of determining the records for inclusion in the list box.

Current Value -- Signifies the value contained in the key field at the beginning of the ACCEPT loop. This is the value used for the range for the duration of the procedure.

Single Value -- Specifies a variable containing the limiting value. Only records matching the variable are included. Enter a variable in the **Range Limit Value** box which appears, or press the ellipses (...) button to select the variable from the Select Column dialog.

Range of Values -- Lets you specify upper and lower limits. Enter a variable in the **Low Limit** and **High Limit Value** boxes which appears, or press the ellipses (...) button to select the variables from the Select Column dialog.

File Relationship -- Lets you choose a range limiting file from a 1:MANY relationship. The Range Limiting field must be the "One" side of a One-to-Many Relationship with the Primary File used in the Process. The relation's linking key must be the same as the Access Key for the procedure. Enter a file in the Related File box, or press the ellipses (...) button to select it from the Select Column dialog.

Hot Fields

You *must* BIND any variables or EQUATEs used in a filter expression. Press Insert to add the variable to the **Hot Fields** list, and check the **Bind Field** box.

Classes

Use the Classes tab to override the global settings for the Class. See Classes Tab.

Procedure Properties--Source

The Source Procedure template provides an elegant and simple way to add hand code to your application. By default, it provides simple source points at which to embed your code: the data section, code section, local procedures and routines.

The template simply declares the procedure, handles any optional parameters, places the embedded data declarations in the data section, begins the CODE section, then places any embedded executable code in the CODE section:

Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.
Prototype	Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator
Module Name	The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Declare Globally	Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.
Export Procedure	Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).
Parameters	Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .
Generate Open/Close Files Routines	Check this box to generate an OpenFiles and a CloseFiles ROUTINE for access in your source procedure. Files that you add in the *Other Tables* entry point in the *Data / Tables Pad* dialog will be automatically processed in these ROUTINES. You will still have to explicitly reference these ROUTINES in your hand code (i.e., DO OpenFiles, DO CloseFiles)

Generate Save/Restore Files Routines	Check this box to generate a SaveFiles and a RestoreFiles ROUTINE for access in your source procedure. Files that you add in the *Other Tables* entry point in the *Data / Tables Pad* dialog will be automatically processed in these ROUTINES. You will still have to explicitly reference these ROUTINES in your hand code (i.e., DO SaveFiles, DO RestoreFiles). The SaveFiles saves a file's position, as well as any active HOLD or WATCH. It also saves a copy of the managed file's record buffer contents.

Procedure Properties--Splash

The Splash Template generates code to display a window with an image and some text on a 3-D bevelled panel. The window closes automatically after a specified amount of time. In addition, you can optionally allow the user to close the window at any time by CLICKING on it.

Frame procedures are designed to optionally call Splash procedures. Alternatively, you can call Splash procedures with embedded source code.

By convention, a splash procedure provides a visual or audio (or both) fanfare for your program. A splash screen can provide a recognizable logo or icon whose familiarity may raise the user's comfort level and may serve as an advertisement for your program. Additionally it diverts the user's attention from the sometimes boring task of loading and initializing the program.

Description A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog.

Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.

Category Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.

Prototype Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator

Module Name The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.

Declare Globally Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.

Export Procedure Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).

Parameters Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .

Return Value For functions, lets you specify the variable receiving the return value.

Window Behavior Press this button to control the behavior of the Window. See Window Behavior

ListBox Styles	Press this button to maintain any List Box Styles used by this procedure. See List Box Styles.
BIND Fields and Procedures	Press this button to maintain all BINDed variables and procedures used in this procedure. See BIND Fields and Procedures.
Display Time (in seconds)	Specifies the maximum amount of time the splash window remains displayed.
Close when the user clicks on the splash window	Checking this box lets the user close the window at any time by CLICKING on it.
INI File Settings	Checking the **Save and Restore Window Location** specifies that a window's location is stored in the application's .INI file, and will open in that position the next time the procedure is called. This is available only if you enable INI File settings in the **Global Properties** dialog.

Procedure Properties
Window, Browse, Form, Viewer, Menu, ViewOnly

This procedure template is the basis for Window, Browse, Form, Viewer, or Menu procedures. Each of these can add other control or extension templates which add buttons or prompts to this window. See the links at the botton of this topic for information on these template prompts.

Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.
Prototype	Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator
Module Name	The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Declare Globally	Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.
Export Procedure	Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).
Parameters	Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .
Return Value	For functions, lets you specify the variable receiving the return value.
Window Behavior	Press this button to control the behavior of the Window. See Window Behavior
ListBox Styles	Press this button to maintain any List Box Styles used by this procedure. See List Box Styles.
BIND Fields and Procedures	Press this button to maintain all BINDed variables and procedures used in this procedure. See BIND Fields and Procedures.

The **Browse** Template adds several control templates including:

Browse Box control template

Browse Update buttons control template

Browse Select button control template

Close Button control template

The **Form** Template adds:

Save Button control template.

Record Validation extension template.

The **Viewer** Template adds:

ASCII View control template

ASCII Search Button

ASCIIPrint Button

CloseButton

The **ViewOnlyForm** Template adds:

CloseButton

ViewFormActions

The **Menu** Template is nearly equal to the default Window template, but adds a default SDI window with a Menu structure defined.

Procedure Properties--Report

This procedure lets you create reports. Press the **Report** button in the **Procedure Properties** dialog to create your report. The procedure template includes a window to show the progress of the report processing. The **Procedure Properties** dialog also includes a checkbox to specify whether you wish to generate a print preview function for your report.

Description	A short text description for the procedure, which appears next to the procedure name in the **Application Tree** dialog. Press the ellipsis (...) button to edit a longer (up to 1000 characters) description.
Category	Use this field to group your procedures in the Category Tab of the Application Tree. The Procedure templates add a default value corresponding to their types to group procedures made with the same template together. You can create your own categories as you like.
Prototype	Lets you optionally type a custom procedure prototype which the Application Generator places in the MAP section. See Prototyping and Parameter Passing with the Application Generator
Module Name	The source code file to hold the code for the procedure. Select from the drop down list. By default, the Application Generator names modules by taking the first five characters of the .APP file name, then adding a three digit number for each module.
Declare Globally	Check this box to generate the procedure's prototype into the PROGRAM's MAP, rather than the MODULE's MAP. This makes the procedure callable from any other procedure, but it also forces a recompile of all program modules whenever you change the prototype.
Export Procedure	Declares the procedure in the export file, enabling it to be called by another application. Note:This checkbox is only available when the target file specified in Application Properties is a Dynamic Link Library (.DLL).
Parameters	Lets you specify parameter names (an optional list of variables separated by commas, with the entire list surrounded by parentheses) for your procedure, which you can pass to it from a calling procedure. You must specify the functionality for the parameters in embedded source code. **See Also:** PROCEDURE Calls .
Return Value	For functions, lets you specify the variable receiving the return value.
Report Behavior	Press this button to control the behavior of the Window. See Window Behavior
ListBox Styles	Press this button to maintain any List Box Styles used by this procedure. See List Box Styles.
BIND Fields and Procedures	Press this button to maintain all BINDed variables and procedures used in this procedure. See BIND Fields and Procedures.

Window Message The default text to display in the Progress window after opening.

Report Properties Press this button to control all report properties provided by the templates. See Report Properties.

Procedure Property Tabs

Report Properties

General

Print Preview Check this box to enable previewing of a report before printing. When checked the **Preview Options** tab will appear.

Data Source Your report can read data to be processed from a target data file (or table), a queue, or from memory.

If you select *File* from the drop list, options for Quick Scan, Record Filter, Additional Sort Fields, Record Count Override, and Progress Bar Limits are available (and discussed below).

If you select *Queue* from the **Data Source** drop list, a **Queue Name** entry is displayed. Press the **E** button to the right of the **Queue Name** prompt to select a target queue that the report will read from. The report reads the queue in record sequence, so it will be important to make sure that the queue is built and sorted properly prior to printing.

Finally, if you select *Memory* from the **Data Source** drop list, the report will *not* process (e.g., loop through) the report file designated in the report procedure's Table Schematic. Essentially, this setting is used to print a "snapshot" of the populated report elements and their values when the print process is called.

If you select Memory or Queue as your data source, you will still have to designate a primary table in the Table Schematic dialog. This table, however, will *not* be processed when these alternative data sources are selected.

MDI Progress Window Check this box to specify that the Progress Window generated for this report will be MDI. This option is necessary if you plan to allow multiple reports to run on different threads at the same time. The disadvantage to this could be a slight reduction in report generation

	performance. DO NOT check this box if your report procedure is not threaded.
Progress Interval Timer	Enter a value in hundredths of a second that specifies the timer interval for this report. The lower the number, the progress update will show a smoother display during report generation. The larger the interval, the less frequent the progress window will be refreshed. Use this setting when you plan to run multiple reports at the same time, and need to fine tune the intervals between report procedures.
Quick-Scan Records	Specifies buffered access behavior for ODBC, ASCII, DOS, or BASIC files. These file drivers read a buffer at a time (not a record), allowing for fast access. In a multi-user environment these buffers are not 100% trustworthy for subsequent access, because another user may change the file between accesses. As a safeguard, the driver rereads the buffers before each record access. To *disable* the reread, enable QUICKSCAN.
Record Filter	Type an expression to limit the contents of the browse list to only those records which match the filter expression. Press the **E** button to call the Expression Editor, to help you construct syntactically correct expressions. This filters all displayable records. When a Record filter is used in conjunction with a Range Limit, only those records within the specified range are filtered. **See Also: Using Range Limits and Filters**
Additional Sort Fields	Type a comma delimited list of fields on which to sort. These sort fields are in addition to the key for the report set in the **Data / Tables Pad**. If no key is specified, this is the sort order used allowing you to sort records without a key.
IPDRV Options	If you are using the IP Driver in your application, press this button to access the IPDRV Options dialog.
Record Count Override	Specify an approximate record count if the report is filtered. The progress indicator uses this value to calculate the percent completed value.
Set progress bar limits manually?	Clear this box to make your procedure read the result set and set the progress bar limits automatically. Setting limits automatically may produce poor performance for some SQL data sets, or erratic or inaccurate progress indicator for unevenly distributed result sets. Check this box to manually provide progress bar limits for the procedure. Setting manual limits can provide faster performance for SQL drivers and more accurate progress indicators for unevenly distributed result sets. This setting is only effective if you specify a Key for the File in

the **Data / Tables Pad**.

Low Progress Bar Limit	Supply the lowest "free" key element value for the result set. You may type the value or the label of a variable containing the value. Enclose literal string values in single quotes ('value').
High Progress Bar Limit	Supply the highest "free" key element value for the result set. You may type the value or the label of a variable containing the value. Enclose literal string values in single quotes ('value').

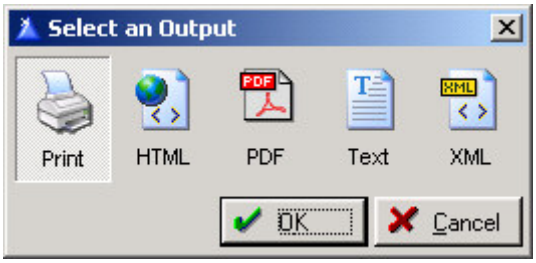
Report Target

This tab control and its options are only available if you have the Advanced Report Generation templates enabled. The options here allow you to designate a default report target and method to use with these supported templates.

Note:

If this tab control does not appear after including the Advanced Report Generation Global , Extension, make sure to refresh the template generation sequence by selecting **Source** or **Embeds** in the proper procedure.

Report Target	From the drop list, select the initial output the report is to be directed to. If you select *Printer*, the report defaults to a standard printed output. Selecting *Other* enables the **Other Target** prompt, where you can select a different output type by default. Select *Ask at Runtime* to allow a popup window to display before the report begins to process. The window displayed at runtime would look something like this:



Add Print like an option	This option is only available if you have selected *Ask as Runtime* as the report target. Check this box to include standard printer output as a user selectable choice at runtime.
Other target	This drop list is only available if you select Other as your Report Target option. Select from HTML, PDF, TXT or XML as your *default* report target.

Note:

If you have more than one "Report To" template set active, you can always change the default target at runtime in the Print Preview window. Simply select **File > Save As** from the menu, and a list of valid report outputs will be displayed for selection.

Hidden Controls

Each Advanced Report Generation template has the option to hide controls that are not applicable to its output. The idea is to have special controls populated on your report for different types of outputs. For example, you may have a field on a report that stores the proper tag information for a PDF document, and in the same area a hot link tag for HTML. In the Report to HTML template, you can hide the PDF Bookmark field, and in the Report to PDF template, hide the HTML specific control.

The **Hidden Controls** button displays a list box of ALL controls hidden for all available formats that you have enabled. This allows you to quickly search for a control that needs your attention.

Range Limits

Range Limit Field

In conjunction with the **Range Limit Type**, specifies a record or group of records for inclusion in the report. Choose a field by pressing the ellipsis (...) button. The Range Limit Field must be a component of the report's Access Key. The range limit is key-dependent; the generated source code uses the SET statement to find the first valid record.

Range Limit Type

Specifies the type of range limit to apply. Choose one of the following from the drop-down list.

Current Value signifies the current value of the Range Limit Field.

Single Value lets you limit the list to a single value. Specify the variable containing that value in the **Range Limit Value** box which appears.

Range of Values lets you specify upper and lower limits. Specify the variable containing the values in the **Low Limit** and **High Limit Value** boxes.

File Relationship lets you choose a range limiting file from a 1:MANY relationship. This limits the report to display only those child records matching the current record in the Parent file. For example, if your report is a list of Orders, you could limit the output to only those orders for the current Customer (in the Customer file).

See Also: Using Range Limits and Filters

Preview Options

The Preview Options tab lets you control the initial appearance of the report preview window. This tab is only available if you check the **Print Preview** box on the General tab.

Tip

Although nearly all font types used on reports will print accurately, for the best display of reports in the Print preview window, use a True Type font (Example – Arial 10pt).

Runtime Skip Preview	Press the **E** button to select a variable or expression that the template will use to determine whether or not a Print Preview window will be displayed prior to printing the report. A TRUE (or non-zero) value will cause the Print Preview to be skipped. A value of zero (or FALSE, the template default) will enable the Print Preview window.
Override Global Report Preview Procedure (Clarion templates only)	Check this box if you would like to create a special print preview procedure to use for this report only. In the **Report Preview Procedure** drop list select an existing procedure, or enter a new name.

The following Preview Options prompts apply to the ABC templates only:

Initial Zoom Setting	Sets the initial magnification for the report to one of seven discrete magnification choices. The end user may change the initial setting at runtime.
Allow User Variable Zooms?	Check this box to let the end user set custom report magnifications in addition to the preset magnification choices.
Set Initial Window Position	Check this box to enable the four following prompts to set the initial preview window position and size.
X Position	The initial horizontal position of the left edge of the window.
Y Position	The initial vertical position of the top edge of the window.
Width	The initial width of the window.
Height	The initial height of the window.
Maximize Preview Window	Check this box to initially maximize the preview window. This supersedes the **Set Initial Window Position**, whose coordinates are applied only when the window is restored to its normal unmaximized state.
MDI Preview Window	Check this box to specify that the Preview Window generated for this report will be MDI. This option is necessary if you plan to allow multiple reports to run on different threads at the same time. The disadvantage to this could be a slight reduction in report generation performance. DO NOT check this box if your report procedure is not threaded.

Breaks

The **Breaks** tab provides support for adding *embedded break logic* that is not associated with any bands or break structures on the report. The two uses for this type of logic are used in totaling and conditional headers and footers. The breaks defined here are solely based on field (column) values, and are not associated with any particular band (as in the traditional breaks).

Press the INSERT button to add a new break. You can also change or delete an existing break, or change the break order by using the appropriate move up/move down buttons.

Each break that is created generates embed points that the developer can use to initialize (*Take Start*), analyze (*Update Total*), and reset (*Take End*) each program break. These embeds are provided by the Break Manager Class, which is the internal logic that supports the template interface. Press the **Embeds** button, and search for *Break Manager* to locate these embed points.

In the Break dialog window, the following prompts are provided:

Name: Enter a descriptive name for the embedded break that you are creating.

New Break Level 0? Embedded breaks can be nested. For example, one break can be a totaling break whose result is used in a second totaling break. Check this box if the break will not be dependent on another break.

Fields Fields are the elements that trigger break logic. Each time the field contents changes, and embedded break is triggered. Additional fields may be added, and *any* field whose contents changes will generate the break.

Totaling Each time that an embedded break is generated, a total result may be calculated. Press the **Insert** button to create a total field that will be calculated when the break is triggered.

Target Field Press the ellipsis button to select a **target field** that will hold the totaling result.

Type From the drop list, select *Count*, *Sum*, *Average, SumPro or Weighted Average* to determine the type of totaling that need to be performed on the break.

Source Field Press the ellipsis button to select a **source field** that will be used as the totaling value.

Source Field (Weight) Press the ellipsis button to select a **source field** that will be used as the additional element in the *SumPro* or *Weighted Average* value as follows:

SumPro += (Source Field * Source Field (Weight))

Weighted Average = SumPro(Source Field * Source Field (Weight))/Sum(Source Field (Weight))

Do Total On Specify that the total will be calculated on *All Records*, or a *Condition*

	that is specified below.
Condition	If your **Do Total On** choice is based on *Condition*, enter an expression here that will only include the **Source Field** value if the condition evaluates to "TRUE".
Reset on Break?	Check this box to reset the **Target Field** after the break has been generated.

Header and Footer

An embedded break can be used to generate a conditional header or footer on a page. An example of this might be a multi-page invoice, where the header of the first page may differ from the header of the second page. In this case, you can create an extra detail section that is used as your "Page Two", and triggers on a page number embedded break.

Print Header?	Check this box if you would like to generate a header detail when an embedded break is detected. Select a detail band to use for the generated **Header.**
Print Footer?	Check this box if you would like to generate a footer detail when an embedded break is detected. Select a detail band to use for the generated **Footer**.

Hot Fields

You *must* BIND any variables or EQUATEs used in a filter expression. Press **Insert** to add the variable to the **Hot Fields** list, and check the **Bind Field** box.

For the use of Hot Fields with embedded breaks, check the **Restore Field on Break** check box. This will store a copy of the hot field value of the previous record read. When an embedded break is detected, the previous record value is restored for printing.

Detail Filters

Each Detail band is listed in the Detail Filters list. To restrict printing of a band, highlight it in the list, then press the Properties button. Provide an expression in the Filter field. The band will only print when this expression is true. Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the prompt.

Optionally, check the **Exclude unfiltered** box to restrict any other detail band which does not have a filter expression of its own.

The **Additional Filters** display will show any filters that are generated by other report support templates (like the ReportChild Extension).

Classes Tab

Use the Classes tab to override the global settings for the Class. See Classes Tab.

IP Driver Options

This dialog is useful if you are using the IP Driver addon product, and is accessible through the **Procedure Properties > Report Behavior > General > IPDRV Options**.

The IP Driver is used for data access and transfer over an IP protocol. More information regarding this powerful addon can be found at the SoftVelocity web site.

Use MRP (Multiple Request Packet)

The IP Driver Version 2.0 introduced the use of Multiple Request Packets (MRP). MRPs are designed to improve the performance of data transfer via IP protocol, by packaging multiple requests (records) into a single IP data packet.

If MRP use is set to ON in a BROWSE procedure, the IP Driver returns a full page of records from the server, instead of one record at a time. In a Report or Process, the MRP is set to the value of RecordsPerCycle. The MRP reduces the roundtrips to the server dramatically. However, if you are doing hand-coded file access for each record read (i.e., NEXT, GET, ADD, PUT, etc.) inside the LOOP structure used to fill the List control, or during a Report or Process Procedure, set this checkbox to FALSE (unchecked).

Formula Editor

Conditional Dialog

A conditional field is a computed field with multiple possible expressions. There are two types of conditional fields--IF structures and CASE structures. The assignment statement executed depends on the evaluation of the IF or CASE condition. For example, an IF structure conditional field called Tax could be 0 if Taxable is FALSE, or Price times TaxRate if Taxable is TRUE.

The Formula Editor allows you to create a conditional expression whose result can then be assigned to a variable. Name your conditional formula in the **Formula Editor** dialog, then press the **Conditionals** button to open this dialog.

Each portion of the expression is edited separately. The components appear in the **Structure** list in the lower portion of the dialog box. Select a component, then edit it in the Statement box. You can add and/or nest IF and CASE structures by pressing the **IF THEN** and **CASE OF** buttons.

Statement A currently selected component (displayed in the **Structure** list) of the actual expression under construction.

Information Describes the currently selected component in the **Statements** box.

Check Tests and validates currently selected component of the expression under construction. A check box appears if it is syntactically correct. An "X" appears if not.

Accept Adds the currently selected component of the expression to the **Structure** list.

Structure Lists the components of the expression in a hierarchical list. Each item selected can be edited separately.

Operators Provides buttons for inserting logical and bitwise operators into the expression. You can also type them in directly.

Data Accesses the **Select Column** dialog, where you can select or define a variable or field as an operand within the expression.

Functions Access a list of built-in Clarion functions in the **Functions** dialog.

User Accesses user defined functions within the application under development, displaying them in the **User Function** dialog.

IF THEN Adds and/or nests an IF THEN structure to the expression.

CASE OF Adds and/or nests a CASE OF structure to the expression.

Delete Deletes the selected statement from the conditional structure.

Formula Dialog

This dialog lists all formulas already created for a procedure, along with their template classes. It allows you to add or edit formulas.

If any formulas already exist for the procedure, this dialog appears when you push the **Formulas** button in the *Application Tree*.

The list displays the following information:

Formula	Lists all existing formulas within the procedure.
Class	Lists the template class associated with the formula. A formula's class determines when its calculation is performed. Each template has its own set of classes. For example, in the Form Procedure Template there is a class called "After Lookups" which tells the Application Generator to compute the formula after all lookups to secondary files are completed for the procedure.
Description	A short text description of the formula.
	The dialog offers the following buttons and their actions:
Select	Loads the currently selected formula into the **Formula Editor** for editing.
New	Loads the Formula Editor, ready to create a new formula.
Delete	Deletes the selected formula.

Formula Editor Dialog

The **Formula Editor** dialog provides access to fields defined in the **Data / Tables Pad**, as well as global or local variables, and facilitates creating syntactically correct expressions.

To create an expression, you press buttons to add components to the Statement line. You can also type in your expression, and check the syntax upon completion.

Name	A descriptive label for the formula.
Class	A formula's class determines when its calculation is performed. Each template has its own set of classes. For example, in the Form Procedure Template there is a class called "After Lookups" which tells the Application Generator to compute the formula after all lookups to secondary files are completed for the procedure. Press the ellipsis (...) button next to the field to view the list of available template classes in the **Template Classes** dialog.
Description	A short text description for the formula.
Result	The variable to which the value of the expression is assigned at run time. Press the ellipsis (...) button next to the field to view the **Select Column** dialog, in which you can select or define the variable.
Statement	The actual expression under construction.
Check	Tests and validates the expression under construction. A check box appears if the expression is syntactically correct. An "X" appears if not.
Information	Describes the currently selected component in the **Statements** box.
Operators	Provides buttons for inserting logical and bitwise operators into the expression. You can also type them in directly.
Data	Accesses the **Select Column** dialog where you can select or define a variable or field as an operand within the expression.
Functions	Access a list of built-in Clarion functions in the **Functions** dialog.
User	Accesses user defined functions within the application under development, displaying them in the **User Function** dialog.
Conditionals	Accesses the **Conditional Dialog**, which allows you to create a conditional expression.

See Also:

How to Create a Simple Assignment Expression

How to Create a Complex Assignment Expression

Functions Dialog

Lists the majority of the Clarion built-in functions. Highlight the function you want, then press the **Select** button.

ABS	AGE	ALL
BAND	BOR	BSHIFT
BXOR	CENTER	CHR
CLIP	CLOCK	DATE
DAY	DEFORMAT	FORMAT
INRANGE	INSTRING	INT
LEFT	LEN	LOG10
LOGE	LOWER	MONTH
NUMERIC	RANDOM	RIGHT
ROUND	SETCLOCK	SETTODAY
SQRT	SUB	TODAY
UPPER	VAL	YEAR

Template Classes

Procedure Setup --Upon Entry into the Procedure

This point occurs immediately after the CODE statement, allowing you to initialize values upon entering a procedure.

Before Lookups--Refresh Window ROUTINE, before lookups

This occurs before any lookups to related records, allowing you to prime any key values needed to perform the lookups.

After Lookups--Refresh Window ROUTINE, after lookups

This occurs immediately after looking up related records, allowing you to use values retrieved from related records in your computation.

Procedure Exit--Before Leaving the Procedure

This allows you to assign values before returning to the calling procedure.

Prime Fields--Prime Fields of the Primary File record at beginning of Insert

Available when a Save Button control template is used, this allows you to pre-assign values to fields when inserting a new record.

Before Filter Check--In Validate Record ROUTINE, Before Filter Code

Available when a BrowseBox control template is used, this allows you to create a formula to be used in the filter expression.

Before Range Check--In Validate Record ROUTINE, Before Range Limit Code

Available when a BrowseBox control template is used, this allows you to assign values before range limits checks are made.

Format Browse--Format a variable in the Browse Box

Available when a BrowseBox control template is used, this allows you to compute values to display in the list box.

Before Print Detail--Before Printing Report Detail

Available only when the Report template is used, this allows you to compute values before a sending a detail structure to a report.

Browse (List Box) Behavior

Scroll Bar Behavior

Alpha distribution

The Alpha distribution divides the scroll bar into 26 segments--one for each letter of the alphabet. Moving the thumb to a segment displays the first record beginning with the segment corresponding letter. For example, moving the thumb to the third segment displays the first record beginning with the letter **c**. If no records begin with **c**, the record with the closest higher value is highlighted.

Last Names distribution

The Last Names distribution divides the scroll bar into 100 segments. Each of these segments is assigned a value based on the distribution of names in an average U.S. telephone book. Each segment is assigned a value and positioning the thumb at that segment displays the first record matching that value. If no records begin with the value, the record with the next closest higher value is highlighted.

The pre-defined Last name distribution is a more accurate method than Alpha when displaying names because names are not evenly distributed alphabetically.

For example, the Browse Box control template uses these values

<null>

ALB

AME

ARN

BAK

BAT

BEN

...

Positioning the thumb on the third segment, highlights the first record beginning with the letters AME. If no records match, the next highest next closest higher value is highlighted.

Fixed Thumb

A fixed thumb positions the thumb (or elevator bar) in the center of the scroll bar. Clicking in the scroll bar above the thumb moves up one page at a time. Clicking in the scroll bar below the thumb moves down one page at a time.

Movable Thumb

A movable thumb positions the thumb (or elevator bar) at the top of the scroll bar when the browse box is initialized. Clicking in the scroll bar above the thumb moves up one page at a time. Clicking in the scroll bar below the thumb moves down one page at a time. Draging the the thumb up or down to a position, highlights the closest matching record for that position in the scroll bar. This is dependant on the type of vertical scroll bar behavior you specify.

Custom

This lets you specify the break points for distribution along the scroll bar. This is useful when you have data with a skewed distribution. Insert the values for each point in the list. This divides the length of the scroll bar into segments. Each value you insert in the list creates a segment. For example, if you specify ten values, the scroll bar is divided into ten segments. Positioning the thumb on the third segment, highlights the first record beginning with the value of the value of the third item you've inserted in the list. If no records match, the next highest next closest higher value is highlighted.

You can use numeric or string constants, or variables. String constants should be enclosed in single quotes (').

Runtime

The Browse Box is initialized and computes the values for 100 break points based on the first and last record in the Range.The distribution points are determined only when the file is opened, therefore there is no performance penalty. The lowest value in the key is subtracted from the highest value to estimate the range of numbers, 100 evenly distributed points in that range are determined and used to control the vertical scroll bar behavior.

Locator Properties

Step

The user types in a single character to advance the cursor bar in the list box to the record that contains the nearest match in the key field. Use this type of locator only when the first Free Key Element is a STRING, CSTRING, or PSTRING. If no free key element is available, the application generator converts a step locator to None.

Entry Locator

An entry box holds the value for the locator. When the end user places a value in the entry box, pressing TAB or reselecting the list box moves the selection to the nearest matching record. If an entry control is not placed in the window, the application generator converts an Entry locator to a Step locator.

If you use the same field more than once as a locator, you must override the default locator. For example, if you have a multi-keyed browse which has an ascending key and a descending key on the same field. To use separate controls (as on separate TABs) for each condition, check the override box and select the second instance from the drop-down list.

Incremental Locator

When the end user types one or more characters, the list box moves the selection to the nearest matching record. Backspace clears the characters, one-by-one, moving the highlight bar to the nearest matching record of the remaining characters. If a STRING control is placed in the window, the characters display, allowing the user to see the characters as they are entered or cleared. If an ENTRY control is placed in the window, the locator works like an **Entry** Locator when the entry control has focus.

If you use the same field more than once as a locator, you must override the default locator. For example, if you have a multi-keyed browse which has an ascending key and a descending key on the same field. To use separate controls (as on separate TABs) for each condition, check the override box and select the second instance from the drop-down list.

Range Limit Prompts

Using Range Limits and Filters

There are many times that you will want to view, process, or report a sub-set of records from a file. There are two ways to do this:

1. Range Limit
2. Filters

Each method has its tradeoffs. Range Limits are much faster to process, but they require that a procedure or control use a limited key as the primary key. Filters are more flexible, since they don't require any special key manipulation, but they are much slower. In any procedure that does sequential processing, you can specify a Range Limit Field, and the type of Range Limit you want to use. The types provided are:

Current Value -Value Limited Keyed Access

The key element specified in the Range Limit Field prompt is the final Fixed Key Element. With this kind of Range Limit, the value of all Fixed Key Elements are saved when the procedure is initialized. These values are used for the duration of the procedure.

Single Value-Value Limited Keyed Access

The key element specified in the Range Limit Field prompt is the final Fixed Key Element. With this kind of Range Limit, the values of all Fixed Key Elements EXCEPT the final Fixed Key Element are saved when the procedure is initialized. The final Fixed Key Element is assigned the value specified in the Range Limit value prompt. This value can be either a variable or fixed value. This value is reevaluated each time a page or entry is loaded or processed.

Range of Values -Ranged Key Access

The key element specified in the Range Limit Field prompt is the first Free Key Element. With this kind of Range Limit, the values of all Fixed Key Elements *except* the final Fixed Key Element are saved when the procedure is initialized. The Low Limit and High Limit Values are used to set the keys for sequential access and to evaluate each record read to insure it is within the valid range. These values can be either fixed values or variables. If variables are used, these variables are reevaluated each time a page or entry is loaded or processed.

There is also support in this dialog for the Higher Key Component, which allows you to specify a secondary key component for **Single Values** and **Range of Values,** and then prime the higher component values accordingly.

Browse Box Only - File Relationship - Value Limited Key Access

All Fixed Key Elements are assigned values as defined in a relationship in the data dictionary. With this kind of Range Limit, it is possible to have multiple Browse Box control templates populated on a window, and as long as the relationships are defined and used, when a parent Browse Box goes out of range, all children (and grandchildren, etc.) Browse Boxes and Controls will automatically be reconstructed.

You *must* BIND any variables or EQUATEs used in a filter expression. Add the variable to the **Hot Fields** list, and check the **Bind Field** box.

Current Value

Signifies the value contained in the key field at the beginning of the ACCEPT loop. This is the value used for the range for the duration of the procedure.

Single Value

Specifies a variable containing the limiting value. Only records matching the variable are included. The variable is reevaluated whenever the window is refreshed. Enter a variable in the Range Limit Value box which appears, or press the ellipses (...) button to select the variable from the **Select a Component…** dialog.

Range of Values

Lets you specify upper and lower limits. Enter a variable in the Low Limit and High Limit Value boxes which appears, or press the ellipses (...) button to select the variables from the **Select a Component…** dialog.

Higher Key Component

The Higher Key Component dialog is used to specify range criteria for key elements that are higher in order than the range key field you have selected as the primary range limit component.

For example, you may have a key defined with a date field and customer number. The key's primary sort (higher component) is the date field, followed by the customer number as the secondary sort. If you choose the customer number as the primary range limit component for your browse, you must also prime the higher key component (date in this example) with a designated value. Failure to do this would cause your range limit to fail on all records, since no date field is zero or NULL.

Based on the key name that you have selected, the higher key component will already be present in this dialog's list box. To modify the default setting displayed, highlight the target component, and press the **Properties** button.

Key Field
This field simply displays the key component you have selected.

Value Type
Choose *Current Value*, *Variable Value*, or *Fixed Value* from the drop list.

Current Value

Primes the key component with the current value stored in the record buffer prior to opening the list box.

Variable Value

Press the ellipsis button next to the **Value** field to assign a variable's value to the selected key component.

Fixed Value

Enter a constant value or expression that will be assigned to the selected key component.

File Relationship

Lets you choose a range limiting file from a 1:MANY relationship. The Range Limiting field must be the "One" side of a One-to-Many Relationship with the Browse Box's Primary File. The relation's linking key must be the same as the Access Key for the Browse Box. Enter a file in the Related File box, or press the ellipses (...) button to select it from the **Select Table…** dialog.

Fixed Key Element

A Fixed Key Element is an element of a KEY that has a fixed value for the displayable records. For example, when setting a range limit of a single value, all key components up to and including the range limit field have a fixed value. Any key components following the Fixed Key Elements are Free Key Elements.

Database Browser

Database Browser Menu Commands

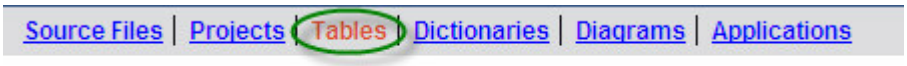
The Database Browser allows you direct access to data files without requiring you to create an application. The only entry constraint is the data type assigned to a column.

The Database Browser offers neither data Validity Checking nor Referential Integrity Constraints. This is a programmer's tool; there are no controls to prevent you from making changes that could compromise the integrity of the database.

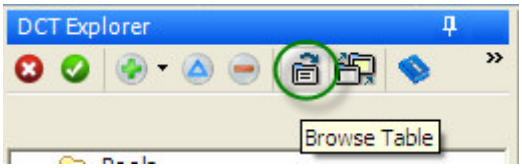
The IDE menu is not associated with any activity of the Database Browser. Some of the menu commands, most notably on the Build and Tools menus, do not specifically reference Database Browser functions. Because the Project System and the Registries are always active, their menu commands are always available.

Access to the Database Browser is accomplished by any of the following actions.

- Using the Start Page **Tables** link:



- From the DCT Explorer, press the Browse Table toolbar button:



- From the IDE Menu, select **Tools > Browse Table**

Database Browser Toolbar

The Database Browser toolbar has three simple options:

Open

Use this option to open additional tables. Each table is opened on a separate MDI tab control.

Save

Press the Save button to save any changes that you have made to the table. As you close the table, changes not saved will be detected and you will be prompted to save any changes.

Tables in an unsaved state are marked with an asterisk (*) on the IDE Tab control.

Display Order

The drop list displays all defined keys of the selected table, and in addition also includes simple **Record Order**. Select an option from the drop list to change the sort order of the active table.

Editing Options

The new Database Browser uses a .NET Data Grid control to display the active file contents.

To edit any column, simply double-click on it. Press the Enter key, or click on another column to complete the edit session. Press the Save button on the toolbar to permanently save the changes.

To delete any record, click on the left most "selection" column on the row to select the entire row, and press the **Delete** key on your keyboard.

To add a new record, scroll down to the bottom of the table, or press CTRL + END on your keyboard. There you will see an empty row where you can add new data.

Find and Filter Options

The new Database Browser also includes advanced locator and filtering options.

Find Options

Filter Options

Find Dialog

The **Find** dialog is used to *locate* the first matching record, based on a simple "*field* = *value*" query. It must find an exact match.

A convenient drop list is provided with a list of valid **Field** (or Column) names.

In the **Field** prompt, select a column from the drop list. Enter a valid search criteria in the **Value** entry. The value to search for must be an exact match. String values should not be quoted.

Filter Dialog

The Database Browser contains a powerful Filter dialog. Use this dialog to range limit a group of records that match your criteria.

You can use the "assisted" dialog (shown above) to filter records based on a simple expression.

If your filter expression is more complex, check the **Free Form** box and entire the desired search criteria.

Some rules when using the free form style:

- If the file has a prefix the Filter expression requires the use of [ ] as in [prefix:fieldname]
- String values on the right hand side of the expression must be enclosed in single quotes.
- LIKE can use either % or * for a wildcard at front or back of the value:
  e.g., LIKE '%test', LIKE '*test', LIKE '%est%'
- The IN operator takes a list of values enclosed in parentheses:
  e.g., LastName IN ('Bagley','Cade'), Zipcode IN (33062, 33358)

Press the **Apply Filter** button to execute the query and display the filtered records in the view window.

Press the **Remove Filter** button to clear all filter values and display all records of the table.

Select File (Database) Drivers

When first loading a file, the Database Browser prompts you to name the driver used to read the file. Select a previously installed Clarion database driver from the list.

This dialog is present whenever adding new file drivers to an existing project, or if you are selecting a table to browse.

The list of registered drivers is displayed. If the driver you are looking for is not on the list, press the **Add Driver** button to register a new one.

After the driver is added to the list press the **Select** button to select the target driver.

See Also: Supported File Systems

Select File Order Dialog

Once a file is open, you can change the sort order by specifying a different key. The drop list at the top of the Database Browser displays a list of available keys, and allows you to change the active key.

Select the key which matches the desired sort order (or Record Order) from the key drop list.

The file is displayed in the selected sort order, and ready for any Database Browser operation.

Dictionary Diagrammer

The purpose of the Dictionary Diagrammer is to provide a graphical representation of an existing data dictionary. Tables, columns, keys and relationships information are provided, and you have the option to control the level of detail. After completing the diagram to your specifications, you can print it, or save it in a variety of image formats. You will be happy with its ease of use and intuitive design.

Here is a quick start guide to using this powerful new IDE tool:

To create a new Diagram:

1. Open the Start Page, and select the **Diagrams** hyperlink.
2. Press the **New** button.
3. When prompted, enter a name for the new Diagram file (DCD extension).
4. When the next dialog opens select the desired Dictionary (DCT extension).

To open an existing Diagram:

You have several options;

From the **File** menu choose the **Open Recent Dictionary Diagrams** menu item.

Or,

Click the Open toolbar button and change the file mask to 'Dictionary Diagram (*.dcd)

Or,

Go to the Start Page, select the **Diagrams** hyperlink, and select the target Diagram to open from the list. or simply press the **Open** button.

General Usage:

There are two list boxes presented; *Dictionary Tables* and *Diagram Tables*

- *Dictionary Tables* represents all tables in the DCT that are NOT on the Diagram.
- *Diagram Tables* represents all of the tables included in the Diagrammer

Incremental locators are implemented for both of these lists.

Together the 2 lists represent all of the tables in the active dictionary. Adding tables to the Diagram Tables list automatically removes it from the Dictionary Tables list(and vice versa).

Clicking on a table in the diagram table should display its properties in 'Table Information' tab.

To add tables to the Diagram

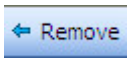
Double-Click on any table in the Dictionary Tables list

OR

Select one or more tables and press the 

button.

To remove tables from the Diagram and Diagram Tables list

Select one or more tables and press the 

button.

To remove tables from the Diagram (not using the list)

Select a single table on the diagram by mouse click , or use the **CTRL + CLICK** or **CLICK** and **DRAG** to select multiple tables, then press the **DELETE** key.

To move tables in the Diagram

CLICK to select a table and then move the cursor over the *Table Title* area, where the cursor changes to the NSEW  shape, and the table is ready to move.

You can also multi-select several tables using **CTRL + CLICK** or **CLICK** and **DRAG** - then move all selected tables.

To resize tables

Click and move cursor to edges, OR

RIGHT-CLICK on an empty space in the diagram for a menu of options and select **Expand** or **Shrink** for all or just selected tables.

Expand means resize to show all fields/keys.

Collapse means make all tables the same size.

Table-names only shows only that.

To Scale the Diagram

Use the Diagrammer toolbar, or

Use the *Pan and Zoom* tool (available on menu)

To Pan the Diagram

The **Alt + Left Mouse** key combination can be used to pan the diagram. Alternatively, holding down the middle mouse button and dragging also will pan the diagram.

To locate/find Tables on the Diagram

DOUBLE-CLICK the table in the *Diagram Tables* list. This will move the diagram so the selected table is visible. This action also causes the **Table Information** tab to display the table's properties.

Diagrammer Toolbar

From left to right:

Open	Not implemented
Save (Diagram or Image)	Save the Diagram file (*.DCD), or save to Bitmap (*.BMP), GIF, JPEG, or PNG
Print (Preview, Print, or PDF)	Preview your diagram prior to printing, print directly, or print to PDF (PDF printing is currently not implemented).
Zoom Out	Make the diagram viewing area larger (effectively shrinking the diagram)
Zoom In	Make the diagram viewing area smaller (effectively enlarging the diagram)
Zoom Percentage	Select from a drop list of specific diagram scaling
Generate SQL	Not implemented at this time. Planned for future release.

Context Menu (on empty Diagram area):

RIGHT-CLICK on an empty space in the diagram for the following context menu:

Auto Arrange Tables	Auto-arrange tries to arrange all the tables using the following rules: 1- The tables are not overlapping each other (z-order) 2- All related tables are moved close to each other 3- Connecting arrows are made as short as possible given the results of rules 1 & 2 If you don't like the layout, you can select it again, and each time it recalculates the diagram layout. Currently, tables with no relations are never moved (e.g., are ignored) by the auto-arrange, so if you move them into the "related tables" area, they will not be moved.

Freeze Table Positions	By default, this menu item is checked ON, and the **Auto-Arrange Tables** menu item is disabled. When this item is checked ON, this means that the position of the existing tables on the diagram are not affected when new tables are added to the diagram. To enable the **Auto-Arrange Tables** menu item, in order to have the Diagram editor position tables for you, toggle the checked state of this menu item to OFF, and then invoke the **Auto-Arrange Tables** menu item.
Collapse All	Make all tables the same preset box size
Collapse Selected	Make selected tables the same preset box size
Expand All	Resize all tables to show all fields/keys
Expand Selected	Resize selected tables to show all fields/keys
Only Fields	Displays just the table and field names.
Only Key	Displays just the table and key names.
Only Table Names	Displays just the table names. This is useful for seeing relations and printing.
Relationship Descriptions	Puts text on the graphic arrows (i.e., *aaaaKey*<-->>*bbbbKey* OR "*Lookup on fieldname"*)
Format	You can control the **Grid Settings** (*Points*, *Lines* or *None*), and **Appearance** of the Diagram from this menu option, including the Background Color and Image, the Tables' Font and Background Colors, and the Arrow Color. You can also select or clear a background image to use in the Diagram.
Trim Document Area	This reduces the width and height of the diagram to just fit all tables. This is extremely useful if you have reduced the size of your diagram, and you need to consolidate the remaining tables into a convenient workspace.

Context Menu (on any Table title bar area):

RIGHT-CLICK anywhere on any table title bar in the diagram for the following context menu:

Arrange from here	Temporarily locks the selected tables and re-positions all other tables. This item is disabled unless a Table is selected (a selected table is shown with white selection handles around the frame).

Table Properties	Opens the Dictionary Editor Table Properties dialog.
Add Field	Opens the Column Properties dialog and allows additional columns.

Working with the Diagram component:

Here are a few tips to consider when working with the Diagrammer.

- Sometimes when a table has many relations or lookups, a table can be drawn on top of another, just click on the Table title area and drag it to the desired location. You can also right-click on an empty area of the Diagram, and from the context menu choose **Auto-Arrange Tables**.
- When you turn on **Relationship Descriptions** from the context menu, description text can intersect the Tables. To resolve this, simply drag the tables to the desired location. Sometimes you may see a row in the table that appears empty, make the table slightly wider to see the text.
- Tables displayed on the Diagram are graphical objects. As such, there are a few things you can do to have a better Diagrammer experience:

  Usually Diagrams of several hundred tables are not very easy to comprehend, and you are usually better off to create multiple Diagrams instead of one huge Diagram.

  If you do intend to add hundreds of tables to a Diagram, then we recommend two things that can make that configuration work more efficiently; first, set the Zoom level to 25%, then add all of the tables at one time. This decreases the number of times the component has to re-arrange table locations to match Parent->Child tables, and reduces the demands on your graphics subsystem to draw the tables. Second, to work with hundreds of tables -- the more graphics horsepower the better!

Caveats in this release:

- Table lists are currently not sorted
- Currently, when closing you are not prompted to save the diagram (but the **Save** toolbar button does save it).
- SQL toolbar button is disabled.
- Open toolbar button is disabled.
- Option to print to PDF is disabled.
- Context Menu (from inside table area) will be hooked to DCT Editor edit forms
- Option to set paper size, etc. for **Print Preview** are not implemented.

Dictionary Diagrammer Options

This dialog allows you to fine tune and tailor many of the display and output options of the Dictionary Diagrammer tool when first opened.

Note:

Again, all of these settings in this dialog apply only to **NEW** diagrams. Once you saved a diagram, all of your settings are saved with it, including for example, the zoom factor.

General-Diagram

Initial Scale (Percentage)
Set the default Zoom percentage here. This setting will be active as the new Diagram is first opened.

Grid Options
Set the grid display techniques. You can specify that the grid is dotted (Points), a solid line (Lines), or neither (None).

General-Tables

Display Relationship Description
Check this box to display the full relationship description.

Display Table Grid Lines
The settings above here control the initial grid type and display settings. You can display points, lines or none in the Diagrammer.

Initial Size for Tables
Expanded displays all elements of the diagram. *Collapsed* displays a condensed list. *Name Only* displays the table name only.

Appearance

These settings control the initial colors displayed for the diagram and table backgrounds. The settings made here will ONLY apply to NEW diagrams that you create. To change existing attributes of a diagram, use the context popup menu provided in the diagrammer.

Diagram Background Color
Select from the drop list a background color to be applied to your new diagrams.

Table Font Color
Set the default font color to display for all textual elements in your new Diagram.

Table Background Color
Your tables can also have an initial background color. Select from the color drop list provided.

Arrow Color
You can control the initial color of your connecting arrows in all new Diagrams created.

Background Image
Press the ellipsis button select a background image for your diagrams. As you select the image, it will be displayed in the Options dialog.

Behavior

Double-Click toggles Expand/Collapse state
If set to ON, a dbl-click on any Table toggles the *Expand* or *Collapse* state of the Keys and Fields nodes. This can be useful when you have all tables collapsed and you want to quickly expand a particular table.

Rearrange Tables when Expanding or Collapsing
When you toggle the *Expand* or *Collapse* state of tables, the default behavior of the Diagrammer will rearrange the layout to either move tables further apart to prevent tables overlaying one another (if Expanding), or move the tables closer together to reduce excess whitespace (if Collapsing). If you already have your tables arranged exactly how you want them, you can set this checkbox to OFF (unchecked), and the tables will *not* be rearranged.

Printing

Print Diagram Background Color
Check this box to print the diagram background color

Print Diagram Background Image
Check this box to print the diagram's background image.

Print Table Fill Color
Check this box to allow the printing of the table background color. Otherwise, table colors will not be printed.

Diagrammer Reports View

The purpose of the Diagrammer Reports View is to give you a powerful visual comparison of key elements in your dictionary. With the Reports View, you can easily locate discrepancies or differences between common dictionary elements across all tables.

To access the Reports View in the Dictionary Diagrammer, press the *Reports View* button located on the Diagrammer Toolbar:

This will open up a series of Summary Data Grid Views of key components of all tables in the active DCT:

Regarding the column headers in the View:

- CLICK and DRAG to reposition.
- CLICK to sort by that column.
- SHIFT + CLICK to add another column to the current sort.
- CTRL + CLICK to remove a column from the active sort.

Press the *Print* button on the Diagrammer toolbar to print the current View.

Note:

Due to a variety of recent upgrades and improvements in the Dictionary Editor, you will need to reconvert your C6 Dictionaries (DCTs) to the new C7 format. Please feel free to discard your previous C7 DCT versions. This is **required** for the proper operation of the new Reports View.

Dictionary Editor

The Data Dictionary is the central repository information concerning your application's data. The Dictionary file--.DCT-- stores file names, file structures, file relations, file aliases and views, field names, lengths, and data types, field validity checks, field entry pictures, keys, indexes, plus much more such as status bar help messages by field and default prompt values by field.

The Application Generator uses the Data Dictionary to generate source code, such as file declarations, which it places in the data section of the generated source code files. It also uses the dictionary to provide, for example, entry pictures when formatting entry dialogs for the end user.

The menu commands available from within the Dictionary Editor are identical to the menu commands of the main IDE. Many dialog in the Dictionary Editor have Help buttons which you can press to view a help topic specifically about that dialog (the F1 key calls the same topic when the dialog is open).

Note that some of the commands, most notably on the Project and Setup menus, do not specifically reference Dictionary Editor functions. Because the Project System and the Registries are always active, their menu commands are always available.

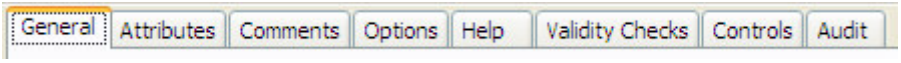
Here are the keys sections of the Dictionary Editor:

Dictionary Explorer

Entity Browser

Dictionary Quick View

New/Edit Column Properties Dialog



The **New Column Properties** dialog allows you to define columns and variables, and to set column or variable related options and attributes.

All the Clarion language attributes that you can place on a column also apply to memory variables. There are only a few additional attributes that can *only* be placed on global or local memory variables. These are disabled when defining a column.

The Dictionary Editor allows you to add the columns one after another, quickly. Each time you complete and close the **New Column Properties** dialog for one column, another blank **New Column Properties** dialog appears, ready for the next column. Press **Cancel** when the blank dialog appears after completing the last column, to return to the **Column/Key Definition** dialog.

General

Column Name
To name the column, type a valid Clarion label in the **Name** prompt. Valid column names may vary slightly according to the file driver.

Derived From
Press the ellipsis button (...) to select another (parent) column in the dictionary from which to copy all column attributes, except the column name. The parent column may be any other column in the data dictionary, including global data, column pool, or file columns.

If a field is derived from another field, then clicking the  button will take you to the derived field.

All fields that have other fields derived from it now have a different icon in the tree display  so you can easily see "derived from" fields.

Fields that have other fields derived from them have a new **Derived Columns** tab in their properties. You can also right-click on these columns and select View Dependant Columns for a similar view. This is a tree showing all fields derived from the field. If a field in this tree will have child fields if there are fields derived from it. Fields in this view have either the normal field icon or a different one (ignore what it looks like at present). The alternate icon indicates that the field has the **Freeze** check box checked.

You can double click on a field in this tree to go directly to its properties.

When you indicate that a field is derived from another field you are now presented with a dialog allowing you to set how the **Freeze** check box is set. If you select the *Always* or *Never* options, you can change the setting later via the **Tools/Options/Clarion/Dictionary Editor Options/Columns** tab.

Change in Behavior for Derived From

In previous versions of Clarion when you changed a field that had other fields derived from it you had to manually apply the changes by either clicking on the Refresh button in the derived field or by selecting Edit/Distribute File. In Clarion 7 changes are automatically applied to derived fields that do not have freeze checked as soon as you save changes to a derived from field.

Because of this behavioral change, when a dictionary is converted from C6 to C7, all fields that are derived from other fields have freeze checked. A message appears in the Dictionary Changes Pad to this effect, but each field is not listed separately.

As the propagation is now automatic, the Freeze check box on Tables and Globals no longer has any meaning. So it has been removed.

You can no longer derive a field from itself. Any dictionaries that have this will have the derived from attribute removed. This change appears in the **Dictionary Changes** Pad.

You can no longer have a field derived from another field with a different type. Any fields that have this will have their derived from attribute removed. This change appears in the **Dictionary Changes** Pad. Once a field is derived from another field, its data type fields are disabled.

Description
To add a text description, type it in the **Description** field. The description appears next to the column name in various dialogs. You can optionally assign the description to the MSG attribute by choosing **Setup** ▸**Dictionary Options** and checking the appropriate box..

Clarion's Wizards use this description when creating Browses, Forms, and Reports.

Data Type
To assign a column type, choose one from the **Type** drop down list. Clarion supports the following column types, which specify how the data will be stored on disk by the file driver, and accessed in memory by the application. These correspond to the Clarion variable types, plus memo and picture types. The types available vary according to the selected database driver. **See Also:** Variable Declaration Statements for a complete list of data types available.

The Decimal type generally provides the best all around performance for mathematical calculations. The compiler optimizes the operation by multiplying values by powers of ten before processing; this greatly speeds up performance on systems without math coprocessors, at no cost in mathematical precision. **See Also:** BCD Operations and Procedures

Base Type
Specify the label of a user-defined datatype. The user-defined data type can be a GROUP, a QUEUE, or an object. See TYPE.

For example:

```
G1 GROUP,TYPE !a user-defined data type
S1  STRING(10)
S2  STRING(10)
   END
```

Use the **Base type** column to specify the G1 label so the Application Generator generates something like the following:

```
MyTypeField G1
```

Reference
To create a reference variable, check the **Reference** box. A reference variable stores the memory address of another variable. This box is enabled only when defining memory variables

Binary
To specify that a MEMO field may hold binary data, check the **Binary** box. This is dependent on the file driver. This adds the BINARY attribute.

Characters
To assign a column length specify a number in the **Chars** field.

Places
To assign a set number of decimal places for a real number, specify a number in the **Places** field.

Dimensions
To declare the variable as an array, and to specify the array dimensions, type them in the **Dimensions** fields. You can specify up to four dimension sizes. This adds the DIM attribute.

Row Picture
To specify the picture for a picture column, type it in the **Record Picture** field.

Screen Picture
To specify a screen picture , type it in the **Screen Picture** field.

"Lock" icon
To lock the screen picture, which specifies that it may not be changed if the column type is changed, press the "Lock" icon next to the **Screen Picture** field.

Prompt Text
To specify the default prompt string, type it in the **Prompt Text** field. The Application Generator uses this for controls which display an on screen prompt.

Column Heading
To specify the default column title, type it in the **Column Heading** field. The Application Generator uses this for reports.

Freeze
Check this box to prevent column attribute refresh from the parent column. Use the **Derived From** field above to set the parent column. See also the **Refresh Column, Refresh Table, Refresh Dictionary,** and **Distribute Column** menu commands.

Attributes

Case
To specify the case attribute for controls referencing the column, choose from the **Normal**, **Capitals** or **Uppercase** radio buttons, in the **Case** group box. The Application Generator adds the CAP or UPR attributes.

Typing Mode
To specify the default typing mode attribute for controls referencing the column, choose from the **Insert**, **Overwrite** or As Is radio buttons in the **Typing Mode** group box. The Application Generator adds the INS or OVR attributes.

Flags

Immediate
To specify immediate event notification for controls referencing the column, check the **Immediate** box. The Application Generator adds the IMM attribute.

Password
To specify the data non-display attribute for controls referencing the column, check the **Password** box. The Application Generator adds the PASSWORD attribute. When an end user types in an entry control referencing this column, the characters typed do not appear on screen.

Read only
To specify the display only attribute for controls referencing the column, check the **Read only** box. The Application Generator adds the READONLY attribute.

Justification
To specify justification for controls referencing the column, select from the **Alignment** drop down list. The Application Generator adds the LEFT, RIGHT, CENTER or DECIMAL attribute.

Offset
To specify an indentation amount for controls referencing the column, specify a number in the **Indent** field. The Application Generator uses this setting as the parameter for the LEFT or RIGHT attribute. The measurement unit depends on the default measurement unit for the window in which a control referencing the column resides.

Initial Value
To specify a default value for the column, type it in the **Initial Value** field. Specifying an initial value for a database column generates an assignment statement.

- A literal character value enclosed in single quotes generates:

```
STATE:Code = 'FL'
```

- For numeric data types, a number without quotes generates:

  CUS:CreditLine = 1000

- The label of a global variable or a local variable that is visible to the procedure generates:

  USER:Preference = MyGlobalPreference

- A function generates:

  INV:Date = TODAY()

Specifying an initial value for a global, module, or local memory variable generates a data declaration statement.

Tip

Functions and variables are valid initial values for database columns, but not for global, module, or local memory variables! Use single quotes to specify literal values for database columns, but no quotes are needed for memory variables.

- A literal character value without quotes generates:

  MyString STRING('LITERAL')

- for numeric data types, a number without quotes generates:

  MyNumber LONG(100)

Note:

Initial Value literals cannot be used for STRING types greater than 1023. Use an embed point to prime values greater than this length.

External Name
To specify an external name for the column, type it in the **External Name** field. This covers cases where the column label within the program is different than the name of the column in the data file; for example, you may be accessing a column through an ODBC connection to a database which allows column names longer than the maximum for a Clarion label. Place the name of the column as it exists in the data table here. **See Also:** ODBC Accelerator Driver

Initialization
(Application Generator Only) Check the **Do not Initialize** box to place the AUTO attribute on the data element defined. This means that the compiler will not initialize the memory area that is reserved for this data element, and it is the programmer's responsibility to do so.

Place Over
To declare the column as an overlay, select another column name from the drop down list. This allows the current column to redefine the other column's location in memory. The Application Generator adds the OVER attribute.

Storage Class
Check box choices are available here based on the scope of where the variable is defined (global, module, or local). By default, a local variable is allocated from stack memory, which means the variable is reallocated for each new instance of the procedure. By default, global or module data is implicitly static.

STATIC

The variable is allocated static memory instead of stack memory, which makes the variable "persistent" from one instance of the procedure to the next.

THREAD

The variable is allocated static memory separately for each execution thread in the program. Thus the value of the variable depends on which thread is executing.

Allocation
Check box choices are available here based on the scope of where the variable is defined (EXE or DLL). By default, a variable is allocated memory in the default program.

EXTERNAL adds the EXTERNAL attribute.

Specifies the variable is defined in an external library, and therefore is allocated no memory by this program.

DLL adds EXTERNAL and DLL (dll_mode).

Specifies the variable is defined externally in a DLL. dll_mode is a switch indicating whether the DLL attribute is active or not. The DLL attribute is required for EXTERNAL variables in 32-bit applications.

Comments

Allows you to enter a text description describing the column. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

Do Not Auto-Populate This Column
Directs the wizards to skip this column when creating Form, Browse or Report procedures.

Population Order
Specifies the order in which the wizards populate columns. Choose **Normal, First,** or **Last** from the drop down list. Wizards populate in this order: all Columns specified as First, then all Columns specified as Normal, and finally all Columns specified as Last.

Form Tab
Specifies the TAB onto which the wizards populate the column. Type the Caption for the TAB or select one you have previously created from the drop down list. This allows you to direct the wizard to group columns in the manner you want.

Add Extra Vertical Space Before Column Controls on Form Procedures
Check this box to direct the wizards to add vertical space between this column's control and the one populated above it.

User Options
The IDE supports 3rd party pre-defined DCT options. These options are stored in *.dctopt* files.

The dctopt file is an XML file with a schema of dctopt.xsd.

Any dctopt file in *<clarion>/data/DictionaryOptions* or *<userdata>\SoftVelocity\Clarion\7.0\DictionaryOptions* directories will be read by the IDE.

The default for SoftVelocity is *<Clarion Root>\data\DictionaryOptions\SoftVelocity.dctopt*.

The IDE will also automatically detect changes in these files and update the list of available options

The text typed into this column is available to any Utility Templates that process this file in the %FieldUserOptions symbol. The individual Utility Templates determine the proper syntax for these user options. See also EXTRACT and %FieldUserOptions in the *Template Language Reference*.

The maximum length for any individual option is 4096 characters.

To add a new entry, press the **Insert** button in the Options toolbar. In the subsequent dialog, you have four choices: *Bool*, *String, Integer* or *Predefined*. Bool is for Boolean type expressions, such as On or Off, True or False, etc.

If you choose **String**, the value has an ellipsis button to open a text window. This allows you to enter a string value.

If you select **Integer**, the user-defined option will be a numeric value.

Choose **Predefined** to get a property that is already defined. This is used to set pre-defined properties for use later. A drop list is provided allow you to choose from the list. Selecting a property from this list not only adds the property but also activates it for use. Follow the instructions provided with your add-on template set.

Follow the instructions provided with your add-on template set.

Help

Help ID
To specify a help ID for controls referencing the column, specify a help topic in the **HLP** field. The Application Generator adds the HLP attribute.

Message
To specify a status bar message for controls referencing the column, type the message in the **MSG** field. The Application Generator adds the MSG attribute. When the control referencing the column has focus, the text appears on the status bar, provided the window in which the control appears has one.

Tool Tip
To specify a popup message for controls referencing the column, type the message in the **Tool Tip** field. When the cursor is idle over the control referencing the column, the message appears immediately below the cursor in a popup box. The Application Generator adds the TIP attribute.

Validity Checks

Choose a validation option in this dialog. The Application Generator uses the information when creating and maintaining controls. When the user completes the field and shifts focus to another control, the application will sound a warning beep and set focus back to the control if the data is not valid.

Allow Null
Check this box to allow a true NULL value to exist in the target column.

No Checks
Specifies no validation. This is the default.

Some validation can still occur, depending on the control you use to display the data, and the attributes of the control. For example, an ENTRY control with the REQ attribute automatically enforces a non-blank entry.

Choices
Type the choices to display in the format "Choice1|Choice2|Choice3." Separate the choices with a pipe (|) character (usually SHIFT+\). The Application Generator adds the FROM attribute to SPIN, LIST, and COMBO controls, or adds text to RADIO controls (see the *Language Reference*).

Values
Type the value to assign when the end user selects the corresponding choice. Type the values in the format "value1|value2|value3." Separate the values with a pipe (|) character (usually SHIFT+\). The Application Generator adds the VALUE attribute to RADIO controls (see the *Language Reference*).

Must be in Numeric Range
Specifies the entry must fall within a numeric range. You may specify a minimum value, a maximum value, or both. The Application Generator generates code to enforce the range you specify, and adds the RANGE attribute to SPIN controls (see the *Language Reference)*.

Lowest
Check this box to set a minimum value, then enter the value in the corresponding spin box. Clear the box to specify no minimum value.

Highest
Check this box to set a maximum value, then enter the value in the corresponding spin box. Clear the box to specify no maximum value.

By entering only a lowest, or only a highest value, you can specify an open ended range.

Must be True or False
Specifies a Boolean entry (yes/no, true/false, off/on). The Data Dictionary Window Control defaults to CHECK.

True Value
Type the value to assign when the end user checks the CHECK control. The Application Generator adds the VALUE attribute to the CHECK control (see the *Language Reference*).

False Value
Type the value to assign when the end user clears the CHECK control. The Application Generator adds the VALUE attribute to the CHECK control (see the *Language Reference*).

Note:

The Application Generator does not generate code to enforce true/false entries because, in Clarion, all entries evaluate as either true or false. This selection affects the default window control in the Data Dictionary and applies the VALUE attribute if the control is a CHECK.

Must be in Table
Specifies the value must match a column in a table. This option is enabled only if you previously related another table or tables. See *Adding or Modifying Relationships* in this chapter.

Table Label
Select the lookup table from the list of related tables. The Application Generator generates code to make sure the entered value is in the selected lookup table.

Tip

Use the FileDrop or FileDropCombo control template in your application to provide a same window pick list for the end user, or use the Actions tab for an ENTRY control to provide a separate window pick list.

Must be in List
Specifies the value must match one of the specified choices. The choices are displayed with a SPIN, LIST, COMBO, or RADIO control.

Choices
Type the choices to display in the format "Choice1|Choice2|Choice3." Separate the choices with a pipe (|) character (usually SHIFT+\). The Application Generator adds the FROM attribute to SPIN, LIST, and COMBO controls, or adds text to RADIO controls (see the *Language Reference*).

Values
Type the value to assign when the end user selects the corresponding choice. Type the values in the format "value1|value2|value3." Separate the values with a pipe (|) character (usually SHIFT+\). The Application Generator adds the VALUE attribute to RADIO controls (see the *Language Reference*).

Note:

The Application Generator does not generate code to enforce Must be in List entries. This selection affects the default window control in the Data Dictionary and applies the FROM and VALUE attributes.

Window

To pre-format a window control referencing the current field, select the **Screen Controls** tab, then specify the options in this dialog.

Tip

By choosing the properties for a control at this time, you can save time later. Every application you generate from the dictionary, and every procedure in the application will automatically format the control the way you want it. If you don't format it here, and if the control requires custom formatting, you will have to custom format it for each procedure and application later.

Select either the prompt or entry field from the **Window Controls** list, then press the **Properties** button. The prompt is the label which appears next to the control. The entry field is the actual control which accepts user input.

Window Controls
In most cases, this list box will show an ENTRY and PROMPT control for the column. Select the control to preformat.

Properties
Allows you to customize the control selected in the **Window Controls** list by displaying its Properties dialog.

Reset Controls
Allows you to return the control type to its default, if you changed it by selecting another from the **Control Type** list.

Control Type
Allows you to select a different control type consistent with the column type and **Validity Checks** selected. For example, if you chose the **Must be in List** option in that dialog, one of the choices will be a list box.

Depending on whether the control can receive focus, (or in the case of a check box, which places the mnemonic in the label), the PROMPT in the **Window Controls** list is deleted.

An IMAGE control cannot receive focus, and also has no PROMPT.

Report

To pre-format a report control referencing the current column, select the **Report Controls** tab, then specify the options.

By choosing the properties for a control at this time, you can save time later. Every application you generate from the dictionary, and every procedure in the application will automatically format the control the way you want it. If you don't format it here, and if the control requires custom formatting, you will have to custom format it for each procedure and application later.

Select the string column from the **Report Controls** list, then press the **Properties** button.

Report Controls
The STRING control for the column.

Properties
Allows you to customize the control selected in the **Report Controls** list by displaying its Properties dialog.

Reset Controls
Allows you to return the control type to its default, if you changed it by selecting another from the **Control Type** list.

Control Type
Allows you to select a different control type consistent with the column type and **Validity Checks** selected. For example, if you chose the **Must be in List** option in that dialog, one of the choices will be a list box.

Depending on whether the type of control, the first string control in the **Report Controls** list is deleted.

Audit

The Audit tab contains read-only information about the selected dictionary element. It includes the date that it was created and llast modified, version number, and user rights to the target element.

DCT Explorer

The DCT Explorer allows you to manage the data tables, aliases, global data and data type pools for your dictionary.

Toolbar Options

Close	Closes the dictionary. If the dictionary is in an unsaved state, a dialog will appear that asks you to save changes.
Save and Close	If the dictionary is in an unsaved state, this option will automatically save and close the dictionary.
Add Table **Add Alias** **Add Global** **Add Pool** **Import Tables**	Pops up a sub menu that lets you add a new table, alias, or import a new table from an external source.

Add Table
The structure represents a FILE. The Application Generator generates a FILE declaration as well as code to read and write the FILE. You can use the columns in the FILE as the parents of derived columns. See Derived From in the Column Properties dialog. Also available from the right-click menu.

Add Alias
Adds a new alias to the active dictionary. An alias creates a second reference for a table without duplicating the file on disk. You can add an alias for a table only if it is already on the Dictionary list, and a Full Path Name exists for the active table. Also available from the right-click menu.

Add Global
The structure represents a group of global data declarations. The Application Generator generates a global data declaration for each column in the structure. You can use the global columns as the parents of derived columns. See Derived From in the Column Properties dialog. This option enables the Generate Last checkbox which causes these declarations to follow declarations made for global variables defined in the .APP. Also available from the right-click menu.

Add Pool
The structure represents a Column Pool. The Application Generator generates no code for this structure. You can use the columns in the pool as the parents of derived columns. See Derived From in the Column Properties dialog. Also available from the right-click menu.

Add Table Category
Add a table category,to allow powerful grouping features to allow subset views of the tables in the Dictionary. Not implemented in this release.

Import Tables
From this option you can import a single table, multiple tables, or tables from another dictionary, dependant on the type of driver selected in the Table Import Dialog.

Change	Allows you to change the options for the selected table or alias. **See Also:** Edit Properties dialog.
Delete	Deletes the selected table or alias. Also available from the right-click menu.
Browse Table	Calls the Database Browser to browse the selected table or alias.
Dictionary Properties	Displays the Dictionary Properties window.
Users	Not implemented at this time.
New Diagram	Calls the Dictionary Diagrammer and loads the active dictionary.
Synchronize	Calls the Dictionary Synchronizer
Import/Export	Allow you to import a textual representation of a dictionary into the current active dictionary. You can import from a pre-C7 text source (TXD), or from the new C7 XML based format (DCTX). You can also export the active dictionary contents to the C7 XML-based DCTX format.

Right Click Menu

Don't forget the right click popup menu. In addition to the items already mentioned above, the following choices are available:

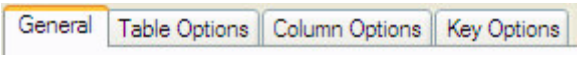
Generate Conversion Program	If you right-click on any table in the DCT explorer, you can create a conversion program using one of three conversion sources; a data file on disk, another file structure in the current dictionary, or another file structure in another dictionary. The next submenu determines the generated code base used for the conversion program. If you have both Clarion for Windows and Clarion.NET installed in the same IDE, this turns into a menu with 2 submenus "using Clarion for Windows" and "using Clarion#" allowing for export of compatible data types to each platforms. This menu item will not appear for SQL tables as data conversion is handled by the server and the generated program is not appropriate to run against an SQL database. The selected table that you right-click on is the *target* table. After optionally renaming or changing the default folder, a second dialog is presented that prompts you to select the *source* or original table to convert. After the program is generated, you will be presented with a dialog option to load the newly generated project into the IDE. In the Tools/Options/Clarion/Dictionary Editor Options/General tab: there is a **Load Conversion Project** option. After a conversion program is generated you can optionally load the project. You can set this to always load the project, never load the project or ask each time a conversion program is generated. The default is to *ask*.

Cut	Copies the file representation to the clipboard, and on **Paste**, removes the table from the Explorer list.
Copy	Copies the file representation to the clipboard, and on **Paste**, does not remove the table from the Explorer list.
Copy Details	Copies the full textual FILE structure to the clipboard. Useful for pasting into custom source. If you have both Clarion.NET and Clarion for Windows in the same IDE, the "Copy Details" context menu for files and fields now turns into a menu with 2 submenus "using Clarion for Windows" and "using Clarion#".
Paste	Retrieves the DCT information from the clipboard.

Dictionary Changes Pad

The Dictionary Changes pad shows all automatic changes that have been applied to the dictionary. You can double click on any item in the pad and that item will be opened in the dictionary editor.

Dictionary Editor Options Dialog



Click on a TAB to see its help

You can customize the default dictionary settings in this dialog. To access the dialog, choose **Tools** ▸**Options** ▸**Clarion** ▸**Dictionary Editor Options**.

The display options in this dialog also affect the appearance of the Data/Tables Pad.

General

Initial Version Text	Sets the text to display to set the version of the dictionary. When you first create a dictionary it has an initial version with a name equal to what you set in that field. During the lifetime of the dictionary you can create new versions. Each version is a marker that is used in the Audit details of an object. If you go to the Audit tab of any object in the dictionary you will see a Version against the Created and Modified details. This indicates which version of the dictionary the object was created in and in with version it was last changed.
Add a Comment to the Dictionary when a new version is created	Check this option to automatically add a comment to the Dictionary every time the version is updated.
Version Comment	The text of the comment to add when auto adding using the option above.
Initial Freeze State	Defines the Initial Freeze State of the dictionary components as they are first imported.
Load Conversion Project	After a conversion program is generated this option sets how you can load the project. You can set this to *always* load the project, *never* load the project or *ask* each time a conversion program is generated. The default is to *ask*.

Enforce Clarion# name restrictions

By default this is on (checked). When checked, the dictionary editor will not allow you to create labels that are not valid in Clarion#. For example. *AS* is an illegal label, but legal in Clarion for Windows. If you do not want to worry about Clarion# restrictions you can uncheck this box.

If Clarion# restrictions are enabled and you load a dictionary into the editor that has a label that violates the Clarion# naming restrictions the editor will automatically rename the object to *ObjectType_OldLabel*, where *ObjectType* is a Table, Alias, Global, Pool, Key or Column. If the object is a Key or a Column with no external name, the external name is set to the old label.

For example, "AS STRING(10)" will be converted to

"Column_AS STRING(10),NAME('AS')

Any such change will appear in the Dictionary Changes pad.

Table Options

Default Driver

To select the default database driver for new tables in a dictionary, choose from the drop down list. For detailed descriptions of the drivers available, see *Database Drivers*.

Default THREAD Attribute

To specify new table definitions default to adding the THREAD attribute (setting aside a separate RECORD buffer for each procedure), check this box.

Default OEM Attribute

To specify new table definitions default to adding the OEM attribute (set International string support), check this box.

Sort Dictionary Tables

Select the *alphabetically* radio button to display the *Tables* list in alphabetical order, *date last modified* to display the tables in the order that they were last modified, and *date created* to display the *Tables* list in the order in which they were created.

Display Table Description

Check this box to display the table description in the Tables list.

Display Table Driver

Check this box to display the table driver in the Tables list.

Display Table Prefix

Check this box to display the table prefix in the Tables list.

Column Options

Display Column Description	Check this box to display the column description in the Columns list.
Display Column Type	Check this box to display the column's data type in the Columns list.
Display Column Picture	Check this box to display the column's display picture in the Columns list.
Display Column Prefix	Check this box to display the column's prefix in the Columns list.
Display Column Derivation	Check this box to display the column's derived column in the Columns list.

Key Options

Display Key Components	Check this box to display the Key components (fields) in the Keys list.
Display Key Description	Check this box to display the Key description in the Keys list.
Display Key Type	Check this box to display the Key Type in the Keys list.
Display Unique Flag	Check this box to display **Unique** if the Key is flagged as unique.
Display Primary Key Status	Check this box to display **Primary** if the Key is flagged as the Primary Key.
Display Other Key Attributes	Check this box to display the other attributes of the Key in the Keys list.
Display Key Prefix	Check this box to display the Key prefix in the Keys list.

See Also:

How to Create a Data Dictionary

Dictionary Properties Dialog

Displays information about the current data dictionary, including creation, modification dates, and a text description.

Audit

Created	The original file creation date.
Last Modified	The most recent modification date. **See Also:** Version ▸ Checkpoint; Version ▸ Revert.

Password

Password options are now located in the Users section of the Dictionary.

Versions

The Dictionary Editor automatically places an internal version number in your dictionary file. A new dictionary begins with version 1.0. You can see the version number/revision number on the caption bar of the Dictionary Properties dialog. This dialog also displays the original creation data and time, and the last modified date and time.

You can do this by going to the Dictionary Properties and selecting the **Versions** tab. From there you can add a new version.

When you add a version the existing dictionary is backed up to *dictionaryName.x.dct* where *x* is the old version number. Optionally a comment is added to the dictionary's comments. The default format of this comment is "**** Version {version} added", where *{version}* is replaced with the version number for the new version.

What comment is added or if a comment is added is controlled in the *Tools/Options/Clarion/Dictionary Editor Options/General* section.

To revert to an earlier version, you can simply rename *dict.1.dct* to *dict.dct* ,and you will go back to version 1.

Comments

Allows you to enter a text description describing the dictionary. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

User Options

The IDE supports 3rd party pre-defined DCT options. These options are stored in *.dctopt* files.

The dctopt file is an XML file with a schema of dctopt.xsd.

Any dctopt file in *<clarion>/data/DictionaryOptions* or *<userdata>\SoftVelocity\Clarion\7.0\DictionaryOptions* directories will be read by the IDE.

The IDE will also automatically detect changes in these files and update the list of available options

The default for SoftVelocity is *<Clarion Root>\data\DictionaryOptions\SoftVelocity.dctopt*.

The text typed into this field is available to any Utility Templates that process this table in the %DictionaryToolOptions symbol. The individual Utility Templates determine the proper syntax for these user options. See also EXTRACT and %DictionaryToolOptions in the *Template Language Reference*.

To add a new entry, press the **Insert** button in the Options toolbar. In the subsequent dialog, you have four choices: *Bool*, *String, Integer* or *Predefined*. Bool is for Boolean type expressions, such as On or Off, True or False, etc.

If you choose **String**, the value has an ellipsis button to open a text window. This allows you to enter a string value. The maximum length for any option is 4096 characters.

If you select **Integer**, the user-defined option will be a numeric value.

Choose **Predefined** to get a property that is already defined. This is used to set pre-defined properties for use later. A drop list is provided allow you to choose from the list. Selecting a property from this list not only adds the property but also activates it for use

Follow the instructions provided with your add-on template set.

Statistics

This tab provides a complete summary total display of all tables, columns, keys, aliases, comments, triggers, and categories defined for the active dictionary.

Dictionary Triggers

The Triggers interface of the Dictionary Editor allows you to enter Clarion source that can be applied before or after any data operation is performed on a selected table.

Note:

In the new Dictionary Explorer, triggers attached to any table are indicated by the following special icon:

A trigger can include any Clarion language statement to execute as a unit and can also invoke stored procedures defined in your application. However, procedures and triggers differ in the way that they are invoked. A procedure is explicitly executed by a user, application, or trigger. Triggers (one or more) are implicitly fired (executed) by the supported database when a triggering INSERT, UPDATE, or DELETE statement is issued, no matter which user is connected or which application is being used.

A **Data** button is also available on this window. The data elements that you add here apply to all trigger types created for the selected table.

Uses of Triggers

Triggers can supplement the standard capabilities of a database to provide a highly customized database management system. For example, a trigger can restrict operations against a table to those issued during regular business hours. A trigger could also restrict operations to occur only at certain times during weekdays. Other uses for triggers are to

- automatically generate derived column values
- prevent invalid transactions
- enforce referential integrity across nodes in a distributed database
- enforce complex business rules
- provide transparent event logging
- maintain synchronous table replicates
- gather statistics on table access
- store complex computed and conditional fields that are table related in one location

Triggers are an extremely powerful tool, but you need to spend some time planning the use of triggers. Consider that when a trigger is executed, if your trigger code accesses another table(s) that a table access statement can cause execution of another trigger(s).

See Also:
Trigger Properties

Dictionary (Client-Side) Triggers

Dictionary Entity Browser

The Dictionary Entity Browser is a tree control that provides a complete view of all elelemnts of your dictionary. Click on any element to display the detail property dialog of the selected element.

This dialog appears as a middle pane within the Dictionary Editor.

Press the **Add** button in the toolbar to add Columns, Keys, Relations, Aliases, and Triggers from this view.

A splitter control separates the Entity Browser from the Properties windows. Drag the splitter bar at the bottom or top to resize, or click on the middle splitter button to hide or unhide the Browser.

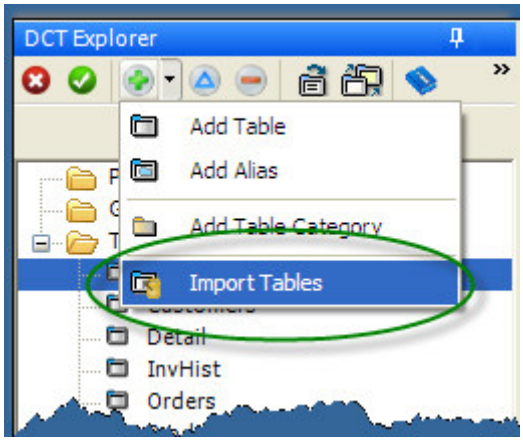
The Dictionary Editor allows you to have multiple tables open at the same time. You can make changes to entities within a table. Pressing the **Save** button within the Entity browser allows you to save the changes to that entity. However, you have the chance to later rollback all your changes by closing the dictionary editor without saving.

So the Save button in the entity editor allows you to save the changes you have made, but still have the option to discard those changes. The Save button on the main IDE toolbar saves the dictionary. At that stage you cannot revert your changes.

Import Table Dialog

The Dictionary Editor allows you to quickly add a data table to the dictionary by creating a data definition based on an existing data file.

With the DCT Explorer dialog active, select **Import Tables** from the Add Button drop list as shown:



The **Table Import Wizard** dialog is displayed. Specify the type of table or dictionary that you are importing from the **Select Server** drop list. Specify the specific table or dictionary in the **Select Dictionary** prompt.

Specify a data table and additional options in the **Select Database** dialog.

Files Type a file name or press the ellipsis (...) button to select a name from the **Open *driver* file** dialog. When importing a definition via an ODBC data source, do not specify a directory name; Clarion will read it from ODBC.INI.

Owner Fill in an optional OWNER attribute.

Driver Options Fill in any optional driver strings.

Note:

After importing any table to the active dictionary, the tables are automatically saved. When you close the dictionary, you will not need to be prompted to save any changes.

See Also: ODBC Accelerator Driver

Insert Table Dialog

Use this dialog to select the table to use for this particular procedure (or control template).

Highlight the table you want, then press the **Select** button.

The tab controls allow you to filter the list of table to select from by sorting by Related Tables only, if you are adding tables to an existing tree.

New/Edit Key Properties Dialog

Click on a TAB to see its help

This dialog allows you to define a key for the currently selected file. **See Also:** How to Create a Key

The Dictionary Editor allows you to add keys and their components one after another, quickly. Each time you complete and close the Properties dialogs for one key, another blank dialog appears, ready for the next. Press **Cancel** when a blank dialog appears after completing the last key, to return to the **Column/Key Definition** dialog.

General

Key Name
To specify a Clarion label for the key, type a valid Clarion label in this field.

Remember that you cannot give a key the same name as one of the fields within the RECORD. One common convention is to use the field name plus the word "key," as in *LastNameKey*.

Description
To place a text description for the key in the Data Dictionary, type it in this field. The description appears in dialogs such as the *File Definition* dialog. If you anticipate using many keys for your application, we recommend filling in this field.

Clarion's Wizards use this description when creating Browses, Forms, and Reports.

Type
To specify a record key, static index file, dynamic index file or Order, choose an option button in the **Type** group. The **Static Index** and **Dynamic Index** options are *disabled* when the **Require Unique Value** box on the Attributes tab is checked, because indexes allow duplicates. **Order** adds a sort order definition to a template symbol (%ORDER) in a similar manner as a KEY without affecting the file declaration.

Note: Orders specified in a dictionary are not yet supported in the included templates. Some third party templates may use this option.

Attributes

External Name
To specify a DOS filename for an external key, type a valid DOS filename in this field.

Require Unique Value
To disallow multiple records with duplicate values in their keys, check this box. This option is valid only for keys, and is disabled for indexes.

Primary Key
To establish the current key as the Primary key, mark this checkbox. The Application Generator adds the PRIMARY attribute. This may be required for certain file drivers.

The primary key must be unique and exclude nulls.

Auto Number
To specify the Application Generator should create code to manage record sequence numbers, check this box.

Case Sensitive
To sort according to case, check this box. When creating or updating the key, all capital letters will precede the lower case letters, as per their positions in the ASCII table.

Exclude Empty Keys
To exclude records with a null or zero value from the key, check this box.

Comments

Allows you to enter a text description describing the key. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

Do Not Auto-Populate This Key
Directs the wizards to skip this Key when creating primary Browse procedures or Report procedures.

Population Order
Specifies the order in which the wizards populate keys. Choose *Normal, First,* or *Last* from the drop down list. Wizards populate in this order: all Keys specified as First, then all Keys specified as Normal, and finally all Keys specified as Last.

User Options
The IDE supports 3rd party pre-defined DCT options. These options are stored in *.dctopt* files.

The dctopt file is an XML file with a schema of dctopt.xsd.

Any dctopt file in *<clarion>/data/DictionaryOptions* or *<userdata>\SoftVelocity\Clarion\7.0\DictionaryOptions* directories will be read by the IDE.

The IDE will also automatically detect changes in these files and update the list of available options

The default for SoftVelocity is *<Clarion Root>\data\DictionaryOptions\SoftVelocity.dctopt*.

To add a new entry, press the **Insert** button in the Options toolbar. In the subsequent dialog, you have four choices: *Bool*, *String, Integer* or *Predefined*. Bool is for Boolean type expressions, such as On or Off, True or False, etc.

If you choose **String**, the value has an ellipsis button to open a text window. This allows you to enter a string value. The maximum length for any option is 4096 characters.

If you select **Integer**, the user-defined option will be a numeric value.

Choose **Predefined** to get a property that is already defined in the designated XML file, located in the \BIN folder. This is used to set pre-defined properties for use later in the associated application. A drop list is provided to allow you to choose from the list. Selecting a property from this list not only adds the property but also activates it for use.

For add-on predefined properties, follow the instructions provided with your add-on template set.

See Also: Using Wizard Options

Columns

Specify the components of keys (the field or fields)-using the Fields tab. You may specify more than one field for a key. Each field is appended to the **Keys** list in the **Field/Keys Definition** dialog.

The **Fields** tab features a list displaying the components.

Sort Order
Choose either the **Ascending** or **Descending** radio buttons to specify the order for the highlighted component.

Note: Not all file drivers support mixing ascending and descending components in the same key.

Insert
Calls the Insert Key Component dialog listing the available fields. DOUBLE-CLICK on the name of a field in the list, to place it in the key.

Delete
Removes the highlighted component from the key.

Move up/Move Down buttons
Moves the highlighted component up or down in the list.

Audit

The Audit tab contains read-only information about the selected dictionary element. It includes the date that it was created and llast modified, version number, and user rights to the target element.

Password Validation Dialog

Allows you to password protect your dictionary, to prevent other developers from modifying it.

To add a password to the data dictionary:

1. From the Dictionary Properties window, press the **Password** button.

2. When the **Password Validation** dialog appears, type a password in the space provided.

3. Press the **OK** button.

See the Development and Deployment Strategies appendix in the *Online User's Guide* for more information.

Dictionary Quick View

Allows you to manage the data tables, aliases, and relations for your dictionary.

Columns List Shows columns for the currently selected table or alias.

Add Column Allows you to add a new column to the selected table shown in the *Tables List*.

Change Allows you to change the options for the selected table or alias. See Also: Edit Properties dialog.

Delete Deletes the selected column.

Use the locator to the right of the **Delete** button to help you quickly locate a table. Useful in large dictionaries.

Move Column Up Moves the selected column shown up in the list.

Move Column Down Moves the selected column shown down in the list.

Related Tables List Shows relations for the currently selected table or alias.

Add Relation Allows you to add a relation to the selected table or alias. **See Also:** New Relationship Properties dialog.

Properties Allows you edit the selected relation.

Delete Allows you to delete the selected relation.

Display Key List Press this icon button to toggle this display to the **Keys List**

Keys List Shows keys for the currently selected table or alias.

Add Key Allows you to add a key to the selected table or alias. **See Also:** New Key Properties dialog.

Properties Allows you edit the selected key.

Delete	Allows you to delete the selected key.

Change Layout	Press this icon button to toggle this display to show/hide the **Relations List**

Other Buttons

Dictionary Properties
Allows you to add or edit information about the current data dictionary, including creation, modification dates, and a text description. **See Also:** Dictionary Properties dialog.

Synchronize
Calls the Dictionary Synchronizer (available only in Enterprise Edition). See Dictionary Synchronizer Overview for more information.

See Also: How to Create a Data Dictionary

New/Edit Relationship Properties Dialog

Click on a TAB to see its help

Set relationships between files in this dialog. The relationships appear in the **Related Files** list on the **Dictionary** dialog, for the currently selected file. When completing this dialog, work from the top down. Start with the **Relationship for** *selected file* group box:

General

Type
Set the relationship type by choosing 1:Many or Many:1 from the drop down list.

Key
Depending on the relationship type selected, choose a primary or foreign key from the drop down list. The choices in the drop down are the keys previously defined for the currently selected file.

Depending on the relationship type you choose for the selected file, the label of the next group box is Parent or Child (respective to 1:Many or Many:1):

Related File

Choose another file from the dictionary to relate to the selected file.

Key

Depending the relationship type you choose for the selected file, the label for this drop down box will be either Primary or Foreign. Select a previously defined key for the related file from the drop down box.

Field Mapping

This group box displays two lists, each showing a key, and the field in the related file which "maps" to it. If the key field names of each file match each other, then just press the **Map by Name** button (below), and Dictionary Editor will automatically link the fields. If they do not, double click on each item in the list boxes, then select a field from the related file that links to the key, in the **Select a Field** dialog.

Map By Name

Automatically defines links based on similarly named fields in each data file.

Map By Order

Automatically defines links based on the order in which fields are defined in each data file.

Alternatively, you can map each field manually by double-clicking the field name in the **Field Mapping** list.

After choosing all other options, set the options in the **Referential Integrity Constraints** group box. The Application Generator automatically generates the code that enforces your selections.

Referential Integrity requires that a foreign key cannot contain any value which has no match in the primary key. This raises potential problems when the end user wishes to change or delete the primary key record.

The **On Update** and **On Delete** drop down boxes each offer the following choices:

No Action

Instructs the Application Generator *not* to generate any code to maintain referential integrity.

Restrict

Instructs the Application Generator to disallow the user from deleting an entry, if the value is used in a foreign key. For example, if the user attempts to change a primary key value, the generated code attempts to check for a related record with the new value, changes it back if necessary, then loops back to the entry dialog so that the user can enter another value.

Cascade

Instructs the Application Generator to update or delete the foreign key record. For example, if the user changes a primary key value, the generated code changes the values in the foreign key that referenced the primary key. If the user deletes a primary key value, the code deletes the foreign key value, too.

Clear

Instructs the Application Generator to change the value in the foreign key to blank or zero.

The following server settings have the same effect as the No Action setting (no code is generated); however, they can provide an indication that RI is enforced elsewhere, and their values can be used to trigger template generated scripts for the server.

Restrict (Server)
Tells the Application Generator that the back-end server will prevent the user from deleting or changing an entry if the value is used in a foreign key.

Cascade (Server)
Tells the Application Generator that the back-end server will update or delete the foreign key record.

Clear (Server)
Tells the Application Generator that the back-end server will set the foreign key to null or zero.

Comments

Allows you to enter a text description describing the relationship. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

User Options

The IDE supports 3rd party pre-defined DCT options. These options are stored in *.dctopt* files.

The dctopt file is an XML file with a schema of dctopt.xsd.

Any dctopt file in *<clarion>/data/DictionaryOptions* or *<userdata>\SoftVelocity\Clarion\7.0\DictionaryOptions* directories will be read by the IDE.

The default for SoftVelocity is *<Clarion Root>\data\DictionaryOptions\SoftVelocity.dctopt*.

The IDE will also automatically detect changes in these files and update the list of available options

To add a new entry, press the **Insert** button in the Options toolbar. In the subsequent dialog, you have four choices: *Bool*, *String, Integer* or *Predefined*. Bool is for Boolean type expressions, such as On or Off, True or False, etc.

If you choose **String**, the value has an ellipsis button to open a text window. This allows you to enter a string value. The maximum length for any option is 4096 characters.

If you select **Integer**, the user-defined option will be a numeric value.

Choose **Predefined** to get a property that is defined. This is used to set pre-defined properties for use later. A drop list is provided allow you to choose from the list. Selecting a property from this list not only adds the property but also activates it for use.

Follow the instructions provided with your add-on template set.

Audit

The Audit tab contains read-only information about the selected dictionary element. It includes the date that it was created and llast modified, version number, and user rights to the target element.

Revert to Previous Revision

This window allows you to rollback a dictionary to a previous version.

Current
Displays the current version.

New Revision
Allows you to select the version to which the dictionary will revert.

Note:

Once you have reverted back to a previous revision, you cannot undo this action.

Select a Column Dialog

Allows you to select a column (from the related table) to link in the **Field Mapping** lists in the **New Relationship Properties** dialogs. Open the dialog by DOUBLE-CLICKING an item in either list.

Columns list	DOUBLE-CLICK or select a column from the related table, and press the **Select** button.
No Link	Allows you to indicate that a column is not part of the relationship.

Select Column Window

The Select Column window presents a relation tree of tables defined in the dictionary. Press the Expand (+) button to display columns attached to a given file, and press the select button to copy the field name to the appropriate entry.

New Table Alias Dialog

Click on a TAB to see its help

An alias creates a second reference for a table without duplicating the file on disk. You can add an alias for a table only if it is already on the Dictionary list.

General

Name	Type a data table "name", as you wish to refer to it in your code. The name must be a valid Clarion label.
Description	Enter a string description for the alias. Clarion displays the descriptions in dialogs such as the **Dictionary** dialog.
Prefix	By default, Clarion will use the first three letters of the Name for the prefix. Optionally specify up to 14 characters to use as a Prefix.
Alias Table	Choose a table from the drop down list. This is the *original* table that the alias "references." The drop down list shows only the tables previously defined using the **Add Table** command in the **Dictionary Properties** dialog.

Comments

Allows you to enter a text description describing the alias. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

Do Not Auto-Populate This Aliased Table
Directs the wizards to skip the Aliased Table when creating primary Browse procedures or Report procedures.

User Options User Options are provided to enable you to provide information to be used by a third-party template set. User Options are comma delimited, that is, each entry is separated by a comma.

Follow the instructions provided with your add-on template set.

See Also: Using Wizard Options

A table alias provides several advantages, at the cost of some system overhead:

- It allows you to set multiple relationships between tables.

  Strict relational database theoreticians state a table may only have a single relational link to another table at a time. Aliases allow you to "legally" work around this limitation. **See Also:** How to Design Your Dictionary and Database

- It allows a second table buffer for the same table.

  You could use this for a second table browse, as well as entry forms and other items for each. This is particularly useful for an MDI application.

- On the negative side, the second table buffer takes up additional memory and resources.

  Any table driver utilizing external key files requires additional file handles for each alias. For example, a table with three external keys and three aliases requires sixteen file handles: one each for the "first" data table and its three keys, and an additional four for each of the aliases. When using aliases, we recommend choosing a database driver that stores keys internally, such as TopSpeed or Btrieve.

- When using aliases, you must open the table in Share mode.

To modify the alias properties at a later time, highlight the table name on the **Dictionary** dialog list, then either double-click or press the **Properties** button.

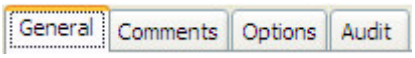
You can edit the columns and keys for the Alias by pressing the **Columns/Keys** button. The Columns/Key Definition dialog lists the columns and keys for the *original* table; any changes you make will update the originals.

New/Edit Table Properties Dialog

This dialog allows you to add a new data table to the list and choose its database driver.

Once the table appears on the list, you may declare columns, keys, set relationships, and other properties for the data table. Using the data from this dialog, the Application Generator will write the FILE structure declaration.

To modify the table properties at a later time, highlight the table name on the **Dictionary** dialog list, then either DOUBLE-CLICK or press the **Properties** button.



General

Label
Type a data table name, as you wish to refer to it in your code. This serves as the label for the Clarion FILE structure. Specify a valid Clarion label--Clarion will automatically truncate the name if necessary. You may also specify a completely different name for the DOS file--see **Full Pathname**, below. **See Also:** Declaration and Statement Labels.

Description
Enter a string description for the table. Clarion automatically displays the descriptions in certain dialogs, allowing you to quickly recognize the table contents.

Prefix
As you enter the data table **Name**, Clarion automatically extracts the first three letters to use as a label prefix when referring to the table. Optionally specify up to 14 characters of your choice in this field. This supplies the parameter for the PRE attribute.

The prefix allows your application to distinguish between similar variable names occurring in different data structures. A column called *Invoice* may exist in one data table called *Orders* and another called *Sales*. By establishing a unique prefix for *Orders* (ORD) and *Sales* (SAL), the application may refer to columns as ORD:INVOICE and SAL:INVOICE.

Driver
Specify the database type. This supplies the first parameter for the DRIVER attribute. When using the Application Generator, Clarion automatically links in the correct database driver library. See *Database Drivers* for a discussion of the relative advantages of each driver. You can specify the default driver by choosing **Setup ▸Dictionary Options**.

Remember that individual database drivers may vary in their support of some of the attributes which you add to the FILE structure in this dialog box.

Driver Options
Optionally type a string for the second parameter of the DRIVER attribute. This conveys additional instructions to the database driver and corresponds to the second parameter for the DRIVER attribute, also known as a "driver string." *Database Drivers* contains additional information. This supplies the second parameter for the DRIVER attribute.

Owner Name
Optionally type a string containing the password for access to the table. This is dependent on the database system. This adds the OWNER attribute to the FILE statement.

You may also check the **Encrypt** box (below) which adds the ENCRYPT attribute. Encrypting the table means that only your application will be able to read the table. It does not mean that it automatically prompts the end user for a password. The end user, however, may not access the data with any other table viewer.

When using the ODBC driver, type the data source name, user ID, and password, separated by commas, in this field.

Full Path Name
Type either the path, or a fully qualified file name for the data table. If you leave the file name out, Clarion automatically uses the first eight letters of the name entered in the **Name** field. You may also omit the table extension--Clarion will supply the correct extension depending on the database driver chosen. This supplies the parameter for the NAME attribute.

When using the TopSpeed driver, if you wish to store multiple tables in a single physical file, separate the file and table names with "`\!`," as in `INVOICE\!ORDERS`. This refers to the ORDERS table in the INVOICE.TPS file.

See Also: TopSpeed Database Driver

When using an ODBC driver to define a FILE such as Microsoft Access, which can store multiple tables in a single file, place the table name in this field. Typically, the name of the physical file which includes the table is listed in the ODBC.INI file; the ODBC driver manager provides this information to the driver.

To specify a variable name for the actual file name, place it in this field, and prefix the variable name with an exclamation point (!).

Create
Optionally specify that the application should create the data table if it does not exist at runtime. This adds the CREATE attribute to the FILE statement.

Reclaim
This option is dependent upon the database driver. It specifies that the application reuse file space formerly taken up by deleted records. Otherwise, the application adds new records to the end of the table. This adds the RECLAIM attribute to the FILE statement.

Encrypt
Optionally turn on table encryption. You must also specify an **Owner Name** (see above). This adds the ENCRYPT attribute to the FILE statement.

Threaded
Optionally specify that each execution thread in your application that uses this table allocates memory for its own separate record buffer. This is typically for use in multiple document applications, and improves table handling. The Clarion default templates automatically add the THREAD attribute on each FILE structure.

OEM
Specifies string data is converted from OEM ASCII to ANSI when read from disk and ANSI to OEM ASCII before writing to disk. This adds the OEM attribute to the table definition.

Bindable
Optionally specify that all variables in the RECORD structure are available for use in dynamic expressions at runtime. The compiler will allocate memory to hold the full Prefix:Name for each variable, instead of using its own internal reference for each variable. Therefore the BINDABLE attribute increases the amount of memory necessary for the application.

Comments

Allows you to enter a text description describing the table. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Options

Do Not Auto-Populate This Table
Directs the wizards to skip this table when creating primary Browse procedures or Report procedures.

User Options
The IDE supports 3rd party pre-defined DCT options. These options are stored in *.dctopt* files.

The dctopt file is an XML file with a schema of dctopt.xsd.

Any dctopt file in *<clarion>/data/DictionaryOptions* or *<userdata>\SoftVelocity\Clarion\7.0\DictionaryOptions* directories will be read by the IDE.

The default for SoftVelocity is *<Clarion Root>\data\DictionaryOptions\SoftVelocity.dctopt*.

The IDE will also automatically detect changes in these files and update the list of available options

The text typed into this field is available to any Utility Templates that process this table in the %FileUserOptions symbol. The individual Utility Templates determine the proper syntax for these user options. See also %FileUserOptions in the *Template Language Reference*.

To add a new entry, press the **Insert** button in the Options toolbar. In the subsequent dialog, you have four choices: *Bool*, *String, Integer* or *Predefined*. Bool is for Boolean type expressions, such as On or Off, True or False, etc.

If you choose **String**, the value has an ellipsis button to open a text window. This allows you to enter a string value. The maximum length for any option is 4096 characters.

If you select **Integer**, the user-defined option will be a numeric value.

Choose **Predefined** to get a property that is already defined. This is used to set pre-defined properties for use later. A drop list is provided allow you to choose from the list. Selecting a property from this list not only adds the property but also activates it for use

Follow the instructions provided with your add-on template set.

Audit

The Audit tab contains read-only information about the selected dictionary element. It includes the date that it was created and llast modified, version number, and user rights to the target element.

Trigger Properties

See Dictionary Triggers for an overview of this powerful feature.

This dialog allows you to enter a trigger for the selected dictionary table. The following prompts are provided:

General

Type

Select the trigger type to designate when the trigger will be executed during one of the following database actions:

(Programmer's Note: Each trigger type corresponds to a related ABC FileManager method)

Type	ABC Method Called
Before Insert	PreInsert
After Insert	PostInsert
Before Update	PreUpdate
After Update	PostUpdate
Before Delete	PreDelete
After Delete	PostDelete

Data

Press the **Data** button to add variables that will be used in your trigger. These data elements will automatically be included in applications that reference the active table. The data elements defined here are only valid for the selected trigger type.

These data items are defined as local data elements of the trigger procedure type (listed above) for the target table. You can only refer to these data elements from within the trigger source text defined, or in the corresponding embed point generated in the Global section of the target application.

Text

Enter the trigger statement in the list box provided by pressing the ellipsis button to enter a full screen editor format.

Make sure to press the **OK** button to save your dictionary trigger after returning from the **Data** or **Text** areas, before attempting to add an additional one.

Comments

Allows you to enter a text description describing the trigger. The description is solely for your convenience, and has no effect on the application. It is useful for situations in which other programmers may pick up your code later, or for when you expect to return to the project after a long period of time since you last looked at it.

Trigger Tokens

To enable the support for Dictionary Triggers in the Clarion Template Family. In the *Global Properties* dialog on the **File Control** tab, check the **Enable Triggers Support** check box.

Also, there is a new token support to control triggers that are specific for both ABC or Clarion chains:

To add code to both chains there are some tokens that can be added.

---TARGET *ChainName*

---ENDTARGET

where *ChainName* can be **ABC** or **CLARION**

Any code outside the tokens will be added to both chains.

For example:

```
!Code for ABC & Clarion
---TARGET ABC
!Code only for ABC
---ENDTARGET
!Code for ABC & Clarion
---TARGET Clarion
!Code only for Clarion
---ENDTARGET
!Code for ABC & Clarion
```

Audit

The Audit tab contains read-only information about the selected dictionary element. It includes the date that it was created and llast modified, version number, and user rights to the target element.

Users Pad

The Dictionary Users pad is opened by the Users button on the DCT Explorer toolbar.

The default user is **Administrator**. This is the primary or master user of the DCT. Press the Change button, and optionally add a password to allow access to this DCT.

You can add multiple users to the dictionary. Users have either read-only or read/write access. If a user has read-only access, they cannot see what other users have access to the dictionary. Users cannot be deleted if they have touched anything in a dictionary. If you attempt to delete a user who has touched the dictionary, you will get a list of all items they have touched.

Summary of User Rules:

- The 1st (default) user is the master user, and can not be deleted, but it can be renamed.
- Any user can create a new user.
- Any user can delete any other users except for the master user
- Users CAN NOT be deleted if they have created any item in the dictionary.
- Users CAN NOT be deleted if they were the last user to modify anything in the dictionary.
- Users marke "read-only" cannot access the user’s information.

Driver String Dialogs

Driver String Builder Dialogs

The Dictionary Editor's Table Properties dialog now includes an ellipsis button next to the Driver String prompt that guides you through the valid options of the selected file format.

The Driver String dialog that will be displayed is dependent on the current File Driver selected. For example, the TopSpeed driver contains only two possible driver string settings, where the Btrieve or MS-SQL dialogs can contain many selections on multiple tab controls.

ASCII Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default ASCII file structure.

Ctrl-Z is End of File Marker

Clip Strings

Buffers

TAB Conversion

End of Record Marker

Quick Scan

BASIC Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default BASIC file structure.

Formatting

Always Quote

Ctrl-Z is End of File Marker

End of Record Allowed within Quotes

Quote Character Code

Comma Character Code

Field Delimiter

End of Record Marker

Data Access

Buffers

Quick Scan

Btrieve Driver String Builder - Creation Switches

The settings on this window will generate an appropriate composite driver string for the default Btrieve file structure definition.

Allow Users to Read Encrypted File

Balanced Keys

Truncate Trailing Spaces

Compress Data

Page Size

Preallocate

Percentage Free Space

Local ACS

ACS File

Btrieve Driver String Builder - Other Switches

Append Buffers

Memo Type

CLARION Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default Clarion 2.1 file structure definition.

Ignore Status

Maintain Header Time

Clipper Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default Clipper file structure definition.

Buffers

Recover

Ignore Delete Flag

Store Zero for 2000

dBase4 Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default dBase4 file structure definition.

Buffers

Recover

Ignore Delete Flag

Store Zero for 2000

dBaseIII Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default dBaseIII file structure definition.

Buffers

Recover

Ignore Delete Flag

Store Zero for 2000

OMNIS

DOS Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default DOS file structure definition.

Buffers

Quick Scan

FoxPro Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default FoxPro file structure definition.

Buffers

Recover

Ignore Delete Flag

Store Zero for 2000

MSSQL Driver String Builder: SQL Generation

The settings on this window will generate an appropriate composite driver string for the default MSSQL file structure definition.

Bind Constants

Clip Strings

Use ODBC CALL Syntax

ORDER in SELECT

Use INNER JOIN

Join Type

Nested Joins

Zero Date or Time is NULL

Extra WHERE Clause

Hint

Auto Incrementing Code

Execute Auto Incrementing code after INSERT

MSSQL Driver String Builder: Finding Tables

The settings on this window will generate an appropriate composite driver string for the default MSSQL file structure definition.

Fast Column Fetch

Gather At Open

Optimize execution of PROP:SQL and opening of SQL Tables

Verify via SELECT

MSSQL Driver String Builder: SQL Communications

The settings on this window will generate an appropriate composite driver string for the default MSSQL file structure definition.

Isolation Level

Bind Column Order

Display Logon Screen

Ignore String Truncation

Save Stored Procedures

Use Trusted Connections to Connect to the Server

Statement Synchronization

MSSQL Driver String Builder: Logging

The settings on this window will generate an appropriate composite driver string for the default MSSQL file structure definition.

Log File Name

Log Message

Allow Details

ODBC Driver String Builder - SQL Generation

The settings on this window will generate an appropriate composite driver string for the default ODBC file structure definition.

Bind Constants

Clip Strings

Use ODBC CALL Syntax

ORDER in SELECT

Use INNER JOIN

Join Type

Nested Joins

Zero Date or Time is NULL

Extra WHERE Clause

Auto Incrementing Code

Execute Auto Incrementing code after INSERT

ODBC Driver String Builder - Finding Tables

The settings on this window will generate an appropriate composite driver string for the default ODBC file structure definition.

Fast Column Fetch

Gather At Open

Optimize execution of PROP:SQL and opening of SQL Tables

Verify via SELECT

ODBC Driver String Builder - SQL Communications

The settings on this window will generate an appropriate composite driver string for the default ODBC file structure definition.

Isolation Level

Bind Column Order

Display Logon Screen

Ignore String Truncation

Statement Synchronization

ODBC Driver String Builder - Logging

The settings on this window will generate an appropriate composite driver string for the default ODBC file structure definition.

Log File Name

Log Message

Allow Details

Oracle Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default Oracle file structure definition.

SQL Communications

Display Logon Screen

Use Asynchronous Calls

Connecting to Personal Oracle 7

SQL Generation

Extra WHERE Clause

Hint

Auto Incrementing Code

Execute Auto Incrementing code after INSERT

Logging

Log File Name

Log Message

Allow Details

Pervasive SQL Driver String Builder: SQL Generation

The settings on this window will generate an appropriate composite driver string for the default Pervasive SQL file structure definition.

Bind Constants Use INNER JOIN

Clip Strings

Use ODBC CALL Syntax Join Type

ORDER in SELECT

Nested Joins

Zero Date or Time is NULL

Extra WHERE Clause

Auto Incrementing Code

Execute Auto Incrementing code after INSERT

Pervasive SQL Driver String Builder: Finding Tables

The settings on this window will generate an appropriate composite driver string for the default Pervasive SQL file structure definition.

Fast Column Fetch

Gather At Open

Optimize execution of PROP:SQL and opening of SQL Tables

Verify via SELECT

Pervasive SQL Driver String Builder: SQL Communications

The settings on this window will generate an appropriate composite driver string for the default Pervasive SQL file structure definition.

Isolation Level

Bind Column Order

Display Logon Screen

Ignore String Truncation

Pervasive SQL Driver String Builder: Logging

The settings on this window will generate an appropriate composite driver string for the default Pervasive SQL file structure definition.

Log File Name and Message
Allow Details

Sybase Driver String Builder: SQL Generation

The settings on this window will generate an appropriate composite driver string for the default Sybase (SQLAnywhere) file structure definition.

Bind Constants Use INNER JOIN

Clip Strings

Use ODBC CALL Syntax Join Type

ORDER in SELECT

Nested Joins

Zero Date or Time is NULL

Extra WHERE Clause

Auto Incrementing Code

Execute Auto Incrementing code after INSERT

Sybase Driver String Builder: Finding Tables

The settings on this window will generate an appropriate composite driver string for the default Sybase (SQLAnywhere) file structure definition.

Fast Column Fetch

Gather At Open

Optimize execution of PROP:SQL and opening of SQL Tables

Force Upper Case

Sybase Driver String Builder: SQL Communications

The settings on this window will generate an appropriate composite driver string for the default Sybase (SQLAnywhere) file structure definition.

Isolation Level

Bind Column Order

Display Logon Screen

Ignore String Truncation

Sybase Driver String Builder: Logging

The settings on this window will generate an appropriate composite driver string for the default Sybase (SQLAnywhere) file structure definition.

Log File Name

Log Message

Allow Details

TopSpeed Driver String Builder

The settings on this window will generate an appropriate composite driver string for the default TopSpeed file structure.

Transaction Control File

DECIMAL Check

Dictionary Synchronizer

Synchronizer Wizard Overview

(Enterprise Edition Only)

The Dictionary Synchronizer synchronizes the open Clarion data dictionary with another Clarion data dictionary (*.DCT), a Clarion text dictionary (*.TXD), or a non-Clarion database such as ORACLE, MSSQL, or Btrieve.

What the Synchronizer Does

Generally speaking, the Dictionary Synchronizer:

- automates the conversion of existing data between different versions of your software;
- automates the synchronization, in either direction, of the "master" data dictionary or database with other project data dictionaries and databases;
- creates a complete Clarion data dictionary, including relationships, from an existing database (SQL, Btrieve, etc.) in a single pass.

More specifically, the Dictionary Synchronizer

- compares a Clarion data dictionary with another Clarion data dictionary or with a database (SQL or Btrieve) to identify any differences between them,
- resolves the differences (with your interactive input, or according to a prior synchronization) by proposing changes to the target dictionary or database,
- validates the proposed changes,
- backs up the target dictionary or (non-SQL) database,
- implements the proposed changes directly to the target dictionary (.dct, .txd, or .ddf), or generates an SQL script to implement the changes to the SQL database,
- generates an SQL script or Clarion program to implement the proposed changes to any existing data,
- saves pertinent information about the process for subsequent reuse.

This functionality aids team development by automating the synchronization, in either direction, of the "master" dictionary or database with other project dictionaries and databases. It lets you easily convert or upgrade existing data to higher or later versions of your software, which is a major benefit to both team and individual developers. Finally, the power of Dictionary Synchronizer applies not only to Clarion data dictionaries, but also to non-Clarion file systems and databases such as Oracle, MSSQL, Scalable SQL, Btrieve, etc., so you can create complete Clarion data dictionaries froma existing databases in a single pass.

Synchronizer Servers

The Dictionary Synchronizer uses Synchronizer Servers to access data dictionaries and databases. A Synchronizer Server is a .dll that communicates with a database or file system such as ORACLE, MSSQL, Btrieve, etc.

Generally speaking, the Synchronizer Server does two things:

1. During the dictionary comparison stage, it queries the dictionary or database and collects a complete, current description of the database's files or tables, keys, views, relationships, objects and properties.
2. During the implementation stage, it generates and optionally executes the programs and database commands necessary to carry out any proposed changes, including converting existing SQL data.

There is a built in Synchronizer Server for Clarion data dictionaries, plus several separate Synchronizer Servers for SQL and Btrieve databases. The Clarion Dictionary server handles all Clarion data dictionaries (regardless of the file driver), and need not be registered.

To register a Synchronizer Server, simply register the corresponding file driver with the Database Driver Registry. See *Clarion's Development Environment—Database Driver Registry* in the *User's Guide* for more information.

Running the Synchronizer

To run the Dictionary Synchronizer, first, be sure to register the appropriate database driver (see *Synchronizer Servers* for more information). Next, open your Clarion data dictionary, then press the **Synchronize** button. This starts the Synchronizer Wizard which leads you step-by-step through the synchronization process. The process requires that you

- identify the *other* dictionary or database with which to synchronize,
- specify the *direction* of synchronization (identify source and target),
- tell the synchronizer *what* to synchronize (all files or some subset of files),
- tell the synchronizer *how to match* files, fields, keys, and relationships (by name, by order, manually, or the same as the last time you synchronized these two dictionaries),
- tell the synchronizer *how to resolve differences* between the dictionaries (copy, delete, ignore, etc.).

The Dictionary Synchronizer does not backup data. We strongly recommend backing up your data prior to any conversion or synchronization.

See Also:

Batch Synchronization

Synchronizer Wizard - Synchronize With

The Dictionary Synchronizer uses Synchronizer Servers to access data dictionaries and databases. A Synchronizer Server is a .dll that communicates with a database or file system such as ORACLE, MSSQL, Btrieve, etc.

There is a built in Synchronizer Server for Clarion data dictionaries, plus several separate Synchronizer Servers for SQL and Btrieve databases. The Clarion Dictionary server handles all Clarion data dictionaries (regardless of the file driver), and need not be registered.

To register a Synchronizer Server, simply register the corresponding file driver with the Database Driver Registry. See *Clarion's Development Environment—Database Driver Registry* in the *User's Guide* for more information.

This window is used with multiple functions available from the Dictionary Editor:

This dialog is used to start the single (SQL based) or multiple (any file system) table import option from your data dictionary. Launched from the **Synchronize** button or **Dictionary Synchronizer** menu item in the Dictionary Editor.

This dialog is also used to begin the stand-alone generation of a multi-table conversion program. Launched from the **Create Conversion Program for Multiple Tables** menu item in the Dictionary Editor.

CLICK on the **Next** button to continue.

Synchronizer Wizard - Select Other Dictionary

The Data Dictionary Synchronizer synchronizes the active dictionary in the Clarion Dictionary Editor with another database or data dictionary.

Press the ellipsis button to select the "other" database or dictionary with the Windows file dialog. If the data dictionary resides in several files (such as a set of Btrieve .DDFs), then select the directory that contains the definition files.

This dialog changes slightly when the Synchronizer is used to *only* generate a multi-table conversion program. You are asked "Which dictionary contains the Old file definitions?". Make sure that the OLD dictionary is the dictionary that contains the SOURCE file definitions to be converted. The NEW dictionary contains the TARGET file definitions to convert to.

For SQL databases (such as ORACLE), the Synchronizer Wizard prompts you for the logon information needed to connect to the SQL database.

CLICK on the **Next** button to continue.

Synchronizer Wizard - Server Login Info

Provide the appropriate connection information for your backend database. This is specific to the type of backend database you are using.

Synchronizer Wizard - Select Previous Session File

The Previous Session file (*.SYN) stores information about the synchronization of two particular dictionaries. Specifically, it stores the files to synchronize from the source dictionary, the files to synchronize from the target dictionary, and a list of matching items (files, fields, keys, and aliases) between the two dictionaries.

If you wish to save the synchronization information, or if you wish to reuse the information from a prior synchronization, you may specify a filename for the Previous Session file.

Press the ellipsis button to select a Previous Session file with the Windows file dialog, or type the Previous Session file pathname.

CLICK on the **Next** button to continue.

Synchronizer Wizard - Select Synchronize Direction

The Data Dictionary Synchronizer never modifies the source dictionary; it only modifies the target dictionary.

Specify whether the open data dictionary is the source or the target by CLICKING the appropriate radio button.

Whenever you consolidate two Clarion dictionaries into a single dictionary, you risk damaging the relationship between the replaced dictionary and its corresponding applications, unless you first export those applications to text (.TXA) format.

CLICK on the **Next** button to continue.

Synchronizer Wizard - Options

These wizard synchronizer options operate exactly the same as the global synchronizer options described in the *Dictionary Synchronizer Options* section. Changing the settings here automatically updates the global settings and vice versa. See the *Dictionary Synchronizer Options* section for more information.

CLICK on the **Matchings** button to tell the synchronizer how to match files, fields, keys, and aliases between the two dictionaries.

Proper matching of dictionary items is essential to producing a useful synchronization.

Synchronizer Wizard - Select files from Clarion Dictionary

Select which files, views, and tables within the Clarion data dictionary to include in the synchronization process. The Synchronizer Wizard provides two file lists for the data dictionary: on the left is a list of files *not* synchronized and on the right is a list of files *to* synchronize. Depending on the database server, the list may show filenames, table names, owner names, etc.

The Synchronizer Wizard lists the files in the order specified on the Options page, that is, alphabetically or according to the source dictionary.

It is to your advantage to include as few files as possible in the synchronization process because fewer files results in a faster process.

Add >
Press this button to move the highlighted file *and all related files and aliases* to the **Files to Synchronize** list. Alternatively, DOUBLE-CLICK the file to synchronize.

< Remove
Press this button to move the highlighted file *and all related files* to the **Files not Synchronized** list.

Add All >>
Press this button to move *all* the files to the **Files to Synchronize** list.

<< Remove All
Press this button to move *all* the files to the **Files not Synchronized** list.

CLICK on the **Next** button to select which files within the other database/dictionary to include in the synchronization process. This page works exactly like the previous file selection page, with one exception: if the other dictionary is a not a Clarion dictionary, loading and updating the file list takes significantly longer than for a Clarion dictionary.

CLICK on the **Finish** button to continue.

Synchronizer Wizard - Matching Options

Proper matching of dictionary items is essential to produce a useful synchronization. The synchronizer provides a range of automatic, intelligent matching choices, which you can combine with manual matchings to quickly and accurately match your dictionary items.

Any items the synchronizer cannot match automatically are flagged within the **Synchronize Dictionaries** dialog so you can manually match them. See *Synchronize Dictionaries Dialog* for more information.

Historically

The **Synchronizer** matches the items based on the synchronizer file you specified earlier. This choice is disabled if you specified a new synchronizer file or no synchronizer file.

By Name Only

The **Synchronizer** matches the items based on their labels and external names.

By Order Only

The **Synchronizer** matches the items based on the order they appear in the two dictionaries.

This option is not available for SQL databases.

By Component Only

The synchronizer matches keys based on the number and labels or external names of the key components.

This option is not available for SQL databases.

The availability of the matching rules depends on the particular database server.

Manually

The **Synchronizer** does not match items. You must match the items with the **Synchronize D**ictionaries dialog.

CLICK on the **OK** button to return to the Options page, then CLICK on the **Next** button to continue.

Pressing the finish button at this point matches the dictionaries according to these settings. For any manually matched items, all sub-items are matched using the settings. For example, if you match a file manually, all its fields are matched by name if By Name Only is specified.

Synchronizer Main Window

When you "finish" the **Synchronizer Wizard**, it opens the **Synchronize Dictionaries** dialog.

Note:

This window is shared between the **Import Tables** and **Synchronize** options found in the DCT Explorer. In the Import Tables function, this window will only appear if there are conflicts with existiong table names in your dictionary. Items in conflict are marked in red.

The **Synchronize Dictionaries** dialog compares the two data dictionaries, item by item and lets you resolve any differences between them. It validates the proposed changes, then generates all the source code, scripts, and data definitions required to implement the changes. Finally, it gives you the opportunity to review and edit the generated source code or SQL scripts.

Overview

The **Synchronize Dictionaries** dialog uses a "file centric" hierarchical list to present the comparison. Beneath each file in the list, the **Synchronize Dictionaries** dialog nests the file's properties and components (fields, keys, relationships, and aliases). Beneath each item in the list, the **Synchronize Dictionaries** dialog nests that item's respective properties and components.

Tip

CLICK on the plus (+) sign to expand a list item; CLICK on the minus (-) sign to contract an item.

For every file, field, key, relationship, alias, and property in the list, the **Synchronize Dictionaries** dialog shows the value from each database or dictionary, plus status indicators to show whether the values are different, and how the difference is resolved (for example, ignore the difference, copy from source to target, delete item from target, not yet resolved, etc.).

The **Synchronize Dictionaries** dialog either dims or omits database properties the target database/dictionary does not support, depending on the settings in the **Synchronizer Options** dialog (see *Dictionary Synchronizer Options*).

Generally, follow these steps to synchronize the dictionaries:

1. Configure the **Synchronize Dictionaries** dialog to sort dictionary items according to your preferences—alphabetically or according to the source dictionary (use the **View** menu).
2. Navigate to each unmatched item and make sure each one is properly matched to the corresponding item in the other dictionary by selecting a matching option from the **Synchronize** menu or the popup (RIGHT-CLICK) menu. Items may be left unmatched if there is no corresponding item in the other dictionary.
3. Navigate to each difference and resolve it by selecting an option from the **Synchronize** menu or the popup (RIGHT-CLICK) menu.
4. Press the **OK** button.

Configuring the Synchronize Dictionaries Dialog

By default, the **Synchronize Dictionaries** dialog displays files, fields, and keys in the order specified in the **Synchronizer Options** dialog. You can also set default colors and other behaviors with this dialog (see *Dictionary Synchronizer Options*). However, you can reset the various sort orders (files, fields, keys) at any time with the **View** menu.

Status Indicators

The **Synchronize Dictionaries** dialog uses status indicators as well as color to provide information about the current synchronization process.

Status indicators are in the right-most column of the list. Status indicators include:

- 32-bit only. Since all applications generated in Clarion 7 are 32-bit, this indicator can be ignored.

- File Structure (field order) changed. This indicates a field's *position* has changed within the target dictionary. This indicator is especially useful when fields are sorted alphabetically and their physical order is not readily visible.


Invalid proposal below. This indicates there is an invalid proposal further down in the hierarchy. Expand the tree or use the navigation controls to find the violation.


Invalid proposal. This indicates the proposed change cannot be accepted by the target server and disables the OK button. For example, number of characters not valid for current data type.

RIGHT-CLICK on this icon then choose Display Validation Error to see an explanation of the violation.

If you have elected to resolve conflicts automatically (see *Dictionary Synchronizer Options*) and the conflict can be resolved automatically, then this status indicator is not set!

Color

Color indicators include:

Blue
indicates no action is applied, and the item is either different than its matching item or it has no matching item.

Black
indicates an action is applied and the resulting target item is valid, or no action is applied but none is needed because the items match and the target item is valid.

Red
indicates an action is applied and the resulting target item is not valid—the target does not support the applied property. You must resolve the invalid proposal before you can complete the synchronization. In the Import Tables function, tables in your existing dictionary that match those that you are importing will be marked in red.

Gray
The item is not supported by the target dictionary.

Navigating the Synchronizer Tree

The **Synchronizer** provides the following navigation aids to let you quickly identify and resolve differences between the two data dictionaries:

Go to the previous difference—automatically expands the list as required. Alternatively choose **Edit ▸Previous Difference**.

Go to the next difference—automatically expands the list as required. Alternatively choose Edit bmc C6H0013.BMP} Next Difference.

Go to the previous invalid proposal—automatically expands the list as required. Alternatively choose Edit ▸ Previous Invalid Proposal.

Go to the next invalid proposal—automatically expands the list as required. Alternatively choose Edit ▸ Next Invalid Proposal.

Matching Unmatched Items

The **Synchronize** menu (and the RIGHT-CLICK popup menu) contains all the functions needed to match items between the two dictionaries.

Match
Associates the selected item with an unmatched item in the other dictionary. The Synchronizer moves the matched items onto the same row. RIGHT-CLICK or press ESC to abort the match.

Unmatch
This uncouples the selected items and moves them to separate rows so they can be rematched. This is for items that were matched inadvertently or which should not be matched.

Match by Name
Unmatches the selected item and rematches it by name.

Match by Order
Unmatches the selected item and rematches it by order of the source dictionary.

Match by Component
Unmatches the selected key and rematches it based on component fields.

DRAG from a source column in one row and DROP on a target column in another row to match the two items. If either item was already matched, this operation first unmatches the original match.

Relationships are matched automatically, based on the files and keys the relationship uses.

Resolving Differences Between Dictionaries

All differences between source and target dictionaries can be resolved by applying an appropriate operations to each item. The **Synchronize** menu (and the RIGHT-CLICK popup menu) contains all the operations needed to resolve differences between the two dictionaries.

The **Synchronize Dictionaries** dialog uses action indicators to provide information about the current synchronization process.

Synchronizer Actions and Action Icons

Action icons are between the source and target columns of the list. Applying an action to any item in the list affects the selected item *and its children*. The action applied to an item cascades to any nested items. The primary item's action icon is in full color; the cascaded action indicators are gray.

Choose **Edit** ▸**Undo** to undo the last action.

The **Synchronize Dictionaries** dialog shows the proposed action for each item in the list. The action icons are in the column between the two dictionaries. Synchronizer actions and their corresponding action icons are:

Icon	**Menu Selection** and explanation of synchronizer action
?	No decision. This is not a menu selection. It is the initial state of the **Synchronize Dictionaries** dialog before any actions are applied. This state does not change the target dictionary.
✚ **Add to target**	Add only. Adds the selected source dictionary items to the target dictionary. This applies only to source items with no matching target items. This action only adds new items in the target dictionary.
➡ **Copy to target**	Change only. Copies the properties of the selected items from the source to the target. This action only affects matched items—it ignores unmatched items. This action only changes existing items in

	the target dictionary.
Merge into target	Add and change. For matching items, this action copies properties from source to target; it adds unmatched source items to the target dictionary; it ignores any unmatched target items. This action can add new items to, and change existing items in the target dictionary. This action makes the target dictionary a union of the source and target.
Replace target	Add, change, and delete. For matching items, this action copies properties from source to target; it adds unmatched source items to the target dictionary; it **deletes unmatched target items.** This action can add new items to, change existing items in, and delete items from the target dictionary. This action changes the target item to exactly match the source item.
Delete from target	Delete only. Deletes the selected items from the target dictionary. This applies only to target items with no matching source items.
Ignore Difference	Ignore. Ignore the selected items. This action does not change the target dictionary.
Set to Default	Parent. Applies the parent's action to the selected items. This action is available only if you have previously changed a child's default operation.

Tip

Choose Edit ▸Undo to undo the last menu selection.

Implementing the Changes

When you have matched all relevant database items, and resolved all differences and invalid proposals in the **Synchronize Dictionaries** dialog, you can implement any proposed changes by pressing the **OK** button.

The **Synchronizer** creates the data conversion program, backs up the target dictionary, and generates the new target dictinary or database definition in the appropriate format—Clarion .DCT or .TXD file, Btrieve .DDF files, or SQL script.

You may execute the SQL script, or you may save the script, optionally edit it, and execute it later. The SQL script both updates the structure of the database and converts existing data to the new structure.

The **Synchronizer** creates back up files by renaming the original target file—**Synchronizer** changes the first position of the file extension to a 'B.' So *.DCT is backed up to *.BCT, *.TXD is backed up to *.BXD, *.DDF is backed up to *.BDF, etc.

Running the SQL Script

How you run the generated SQL script depends on the SQL server. See your SQL database documentation for instructions on storing and running scripts.

Synchronizer Validation Error Window

This window provides an explanation of the validation error. This error must be resolved before you can synchronize the dictionaries.

Synchronizer Options Dialog

The Dictionary Synchronizer is configurable. That is, to some extent you can determine how the synchronizer looks and acts. The main synchronizer elements you can configure are

- colors,
- sort sequences,
- automatic generation of data conversion program,
- automatic resolution of conflicts.

The default settings are suitable for many cases so you probably don't need to change these settings at first. However, at some point you may want to change the default settings.

To reconfigure the synchronizer, choose **Setup ▸Dictionary Synchronizer Options**.

You can change many of these settings with the View menu while the Synchronize Dictionaries dialog is open.

Batch Synchronization

For every synchronization, the Dictionary Synchronizer creates a log called synlog.txt in the root directory of the current drive (root:\synlog.txt). This log serves two purposes: it can be used to rerun a synchronization in batch mode, and it should be submitted with any bug reports along with the synchronized data dictionaries to aid in the debugging process.

After you finish a synchronization (either successfully or unsuccessfully), you can use this log file to automatically rerun the synchronization in batch mode. This can save lots of time for large or repetitive synchronization tasks.

To Synchronize in batch mode:

1. Copy or rename the root:\synlog.txt to another location (otherwise the next synchronization will overwrite it).
2. Use the copy of synlog.txt as the first parameter in the RerunSynchronizer DDE command (see example program).

The second parameter is the Clarion dictionary that started the original synchronization—not necessarily the source dictionary.

3. Run the example program.

The synchronizer starts and repeats the steps logged in the synlog.txt file. Note: If you pressed OK or Cancel at the end of a synchronize, you may want to delete those commands from the synlog.txt file.

```
BatchSynchronize   PROGRAM

DDEChannel      LONG
DDEErrorMsg     CSTRING(300)
DDEActiveDct    CSTRING(300)
DDEErrorNum     USHORT
BatchLog        STRING('c:\SyncTutr\batchlog.txt')
ClarionDct      STRING('c:\SyncTutr\sample.dct')

  MAP
    INCLUDE('DDE.CLW')
  END
  CODE
  System{PROP:DDETimeOut} = 12000     !Time out after two minutes
  DDEChannel = DDEClient('ClarionWin')
  IF DDEChannel < 1
    RUN('CW')
    LOOP
      DDEChannel = DDEClient('ClarionWin')
      IF DDEChannel < 1 THEN CYCLE.
    END
  END
  DDEExecute(DDEChannel,'[RerunSynchronizer('&BatchLog&','&ClarionDct&')]')
  DO CheckDDEError
  DDEClose(DDEChannel)

CheckDDEError   ROUTINE
  DDEErrorMsg = ''
  err# = ERRORCODE()
  IF err# > 600
    ! DDEExecute Failed
    IF err# = 603
      DDERead(DDEChannel, DDE:manual, 'GetErrorNum', DDEErrorNum)
      DDERead(DDEChannel, DDE:manual, 'GetErrorMsg', DDEErrorMsg)
      MESSAGE('Error ' & DDEErrorNum & ' : ' & CLIP(DDEErrorMsg))
    ELSIF err# = 605
      MESSAGE('DDE timeout')
    ELSE
      MESSAGE(err#)
    END
  END
```

File Import Window

The File Import dialog is used to enter the required connection information to a data source that you are attempting to import into your data dictionary.

Connection
Enter or paste a valid connection string to use to connect to your data source. Press the ellipsis button to call the Microsoft Data Link interface, which is used to construct a valid connection string for you.

User ID
If your data source requires a User ID to connect, enter it here, or select a previously used User ID from the drop list provided.

Password
If your data source requires a valid password to connect, enter it here, or select a previously used Password from the drop list provided.

Include/Exclude System Files
Check the appropriate radio button to include or exclude all system files associated with the data source. Normally, system files are administrative in nature and should be excluded from your import.

Select Table to Import
With a single table import, you will be presented with a list box of valid import tables to choose from. Highlight the table you wish to import, and press the Finish button to complete the import process

Dictionary Synchronizer Options

Sort files alphabetically

Checking this box displays files and aliases in alphabetical order on the **Synchronize Dictionaries** dialog. Clearing the box displays them in the order they are defined in the source database.

You may also specify the display order during synchronization by choosing **View ▸Sort files alphabetically**.

Sort fields alphabetically

Checking this box displays fields in alphabetical order on the **Synchronize Dictionaries** dialog. Clearing the box displays them in the order they are defined in the source database.

You may also specify the display order during synchronization by choosing **View ▸Sort fields alphabetically**.

Sort keys alphabetically

Checking this box displays keys in alphabetical order on the **Synchronize Dictionaries** dialog. Clearing the box displays them in the order they are defined in the source database.

You may also specify the display order during synchronization by choosing **View ▸Sort keys alphabetically**.

Create conversion program

Checking this box tells the synchronizer to automatically generate a Clarion program to convert existing data to the new format of the target database.

This box is disabled when the target is an SQL database, because no separate conversion program is required. Rather, the synchronizer generates an SQL script to both change the database definition and convert existing data.

Hide unsupported attributes

Checking this box tells the synchronizer to hide (omitted from the **Synchronize Dictionaries** dialog) any attributes of the database the synchronizer server does not support. Clearing the box tells the synchronizer to display the unsupported attributes, but as disabled (dimmed) items.

Correct validity violations automatically

Checking this box tells the synchronizer to automatically resolve conflicts where possible. Clearing the box tells the synchronizer to display the Invalid Proposal icon so you can resolve the problem manually.

For example, a field data type is record picture, but the target dictionary does not support record picture, so the synchronizer automatically resolves the violation by applying the same operation on record picture as on data type.

The synchronizer cannot automatically resolve some conflicts, such as maximum number of MEMO fields exceeded.

If you have elected to resolve conflicts automatically and the conflict can be resolved automatically, then the unresolved difference status indicator in the Synchronize Dictionaries dialog is not set!

Generate DBA instruction script

Use this option when synchronizing to an existing SQL database. Checking this box tells the synchronizer to generate instructions for *modifying* an existing SQL database. The instructions are an approximation of the SQL needed to achieve the change.

Resolving Invalid Proposals

Invalid proposed changes to the target dictionary can be resolved in two ways: from within the Synchronizer and from outside the synchronizer.

Within the Synchronizer, choose **Display Validation Error** from the menu to find out exactly what the problem is. **Display Validation Error**
Provides information about the nature of the invalid proposal—describes why it is invalid.

If you don't need the offending item in the target dictionary/database, you can proceed with the synchronization by unmatching the item (RIGHT-CLICK the item then choose **UnMatch** from the popup menu), then deleting it from the target (RIGHT-CLICK the item then choose **Delete** from the popup menu).

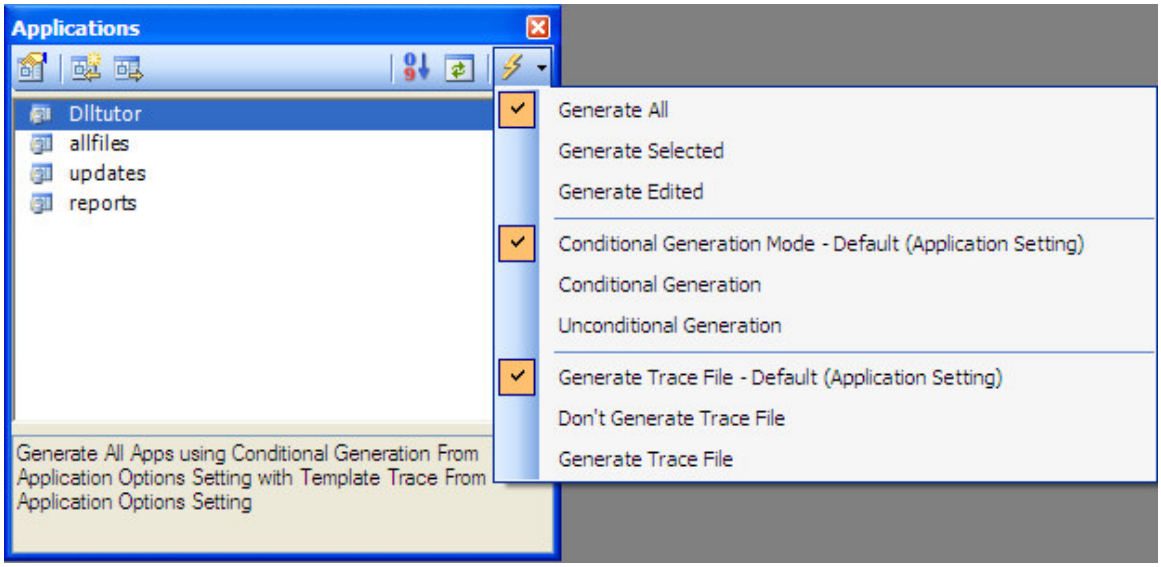
If you do need the item in the target, you may be able to resolve the violation by changing the Synchronizer operation applied to the item. For example, if you copied the item from the source to the target (you chose **Copy to Target** from the menu), then all properties of the source item (including any unsupported properties) are applied to the target item, resulting in an invalid proposal. Changing the operation from "copy" to "ignore" should resolve the problem (RIGHT-CLICK the item then choose **Ignore Differences** from the popup menu) because the Synchronizer does not copy the source item properties to the target item.

Alternatively, you may resolve the violation by canceling the synchronizer session, changing the offending data declaration in the source dictionary/database, then resynchronizing.

Pads and Views

Applications Pad

The Applications Pad provides a view of all opened applications in the IDE. You can also manage your applications with regards to source generation, and compiling and building the project all from this pad.



The toolbar provides the following options:

Edit Application
Opens the Application Editor for the selected application

Create Application from TXA file
Opens a utility that allows you create an application from an Application Text file (TXA).

Export to Text
Exports the selected application to a TXA (Application Text) file.

Sort by Generation order
Toggle the application sort order between generation order and alphabetical.

Refresh
Refreshes the application pad. Usually this is required after a project import or conversion.

Generation Menu Options

To the far right of the pad is a drop menu with a number of generation options. All of these items are toggle settings divided into three sub categories. Pressing the toolbar button will then execute generation based on the current drop menu settings.

Generate All
Generates source for all applications opened for editing (those opened in the Application Editor).

Generate Selected
Generates source for the selected application only. You can multi-select applications in this Pad if needed, and this option will generate all applications selected.

Generate Edited
Generates source for the application currently being edited and selected (on top) in the Application Editor.

Conditional Generation Mode

Choose from three options. Use the default setting as defined in the **Application Options** dialog, or override those settings by switching to **Conditional** or **Unconditional** Generation..

Generate Trace File

Source Generation optionally includes a trace option as defined in the **Application Options** dialog. You can also override those settings by switching to **Generate** or **Don't Generate** as needed. This file should be use only for diagnostic purposes, as it can sometimes affect generation performance.

"Opened" applications are applications that are either loaded in the Application Editor, or, they can be applications that are "lazy loaded", meaning that the application registry and any parsing of source libraries required by the templates has been performed.

Open (or "lazy loaded") applications in the Applications Pad are identified by the following icon:

Applications not opened (or "lazy loaded") are identified by the following icon.

If you select an application to generate, and the application is not already opened, the Generate action will "lazy load" the application as described above.

Generate All
Generates source for all applications listed in the Applications Pad. Applications not marked as opened will be "lazy loaded" as described above.

These "generate" buttons can also be found on the Main IDE Toolbar.

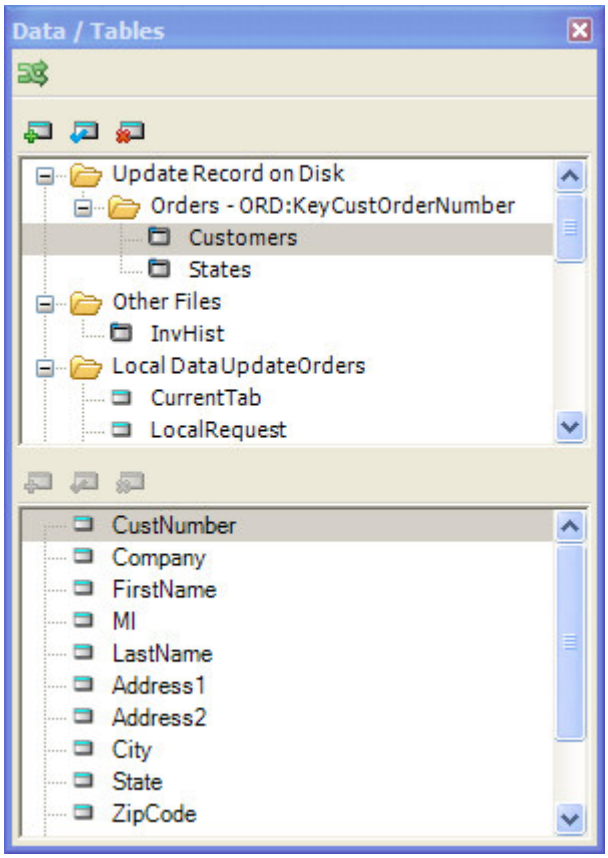
Data Sources View

The Data Sources View displays a hierarchal tree of files, queues, groups and other structures detected in the current active project. This pad is used to quickly populate components in windows and reports.

Simply drag and drop the desired data element from this view to the target window area.

Data / Tables Pad

The **Data / Tables Pad (DTP)** is a special pad that displays the contents of an active dictionary and other application data elements (Global, Module, and Local). This pad is available when either the Dictionary Editor or Application Generator is opened.

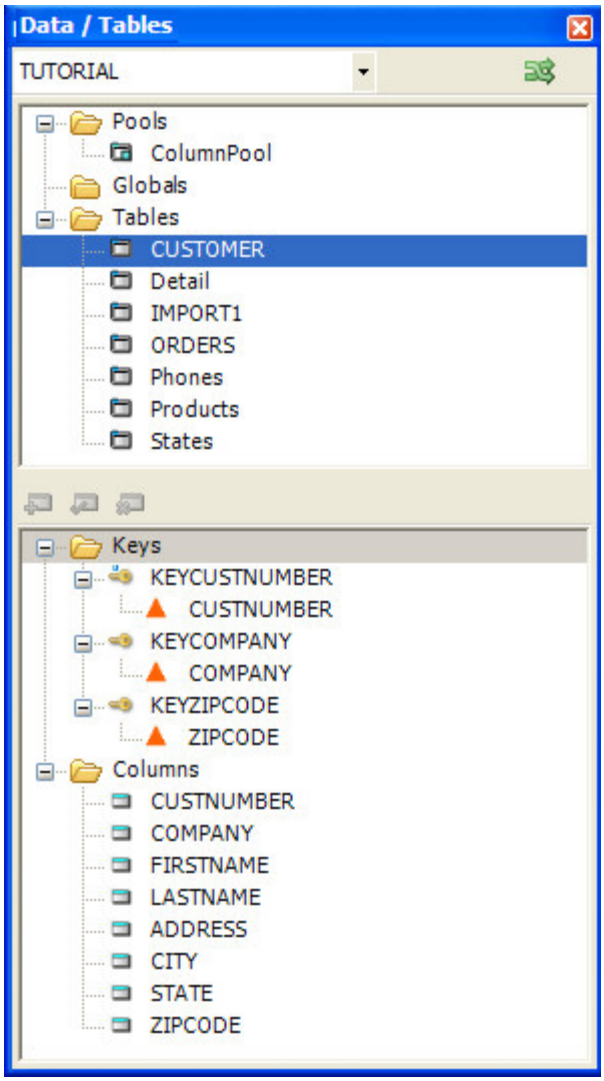


Tables and application data is displayed and maintained from a single source pad. Press the F12 key at any time to open (or bring focus to) the DTP in the IDE. Buttons at the top of the DTP lists allow you to add, modify, or remove data elements where appropriate.

When the Application Editor is active, data in the DTP that is disabled indicates that the global data source is from the global pool of the active dictionary, and may not be deleted.

In the Window and Report Designers, drag a field (column) from the DTP to the window or report area for easy and quick population. The **Populate Columns** and **Populate Multiple Columns** utilities that you used in earlier version are also available in Clarion 7 if you would prefer to use them instead. Access these utilities from the Window or Report Designer menus.

You can also view the DTP from the Dictionary Editor as an alternative view of your dictionary schema.



Note:

To change the display options of the Data/Tables Pad, see the Dictionary Options dialog. (From the IDE Menu, **Tools > Options > Clarion > Dictionary Editor Options**).

Dictionary Changes Pad

The Dictionary Changes pad shows all automatic changes that have been applied to the dictionary. You can double click on any item in the pad and that item will be opened in the dictionary editor.

Classes View

The purpose of the **Classes** View is to display all code and data objects in use by the active project. It also serves as an easy way to locate the target source module where the selected object is defined.

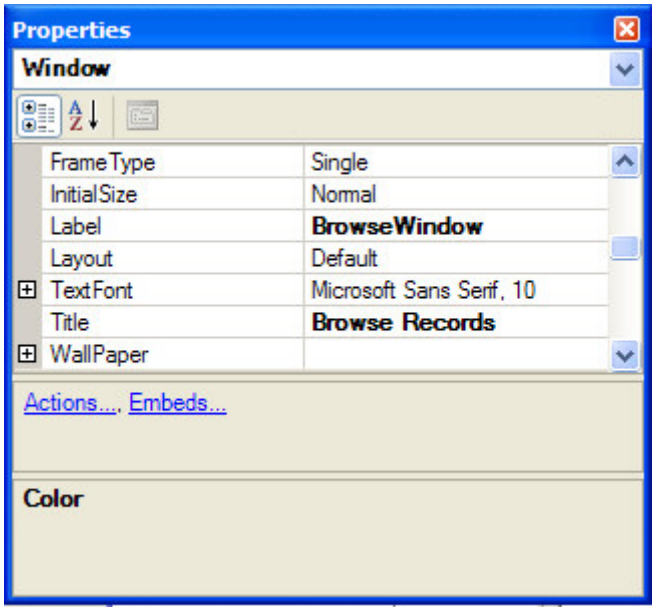
In the toolbar, use the arrow keys to traverse between class branches. There is also a drop list available in the toolbar where you can limit the tree to the type of objects that you wish to display.

In the tree list, you can right-click on any object, and **Rename** any class element or **Find References** where that class member is referenced in source throughout the entire project.

Window Designer - Properties Pad

The **Window Designer's Properties** Pad allows you to quickly specify the appearance and content of the text on each control within the window and on the window title bar. Control the font, size, style, and content of all your text, using standard entry fields and drop down lists. Every property of a selected control is listed in this pad.

Display and get focus to the **Properties** Pad by choosing **View ▸Properties,** or press the F4 key**.** Resize the **Properties** Pad by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG. Use the toolbar buttons to change the sort appearance.



As you multi-select controls in the Designer (CTRL + CLICK), any change in a property in the Properties Pad will affect all selected controls.

You can manually enter in property values in the pad, or clicking on the property name will cycle through valid choices where applicable. In many properties, there are lookup buttons that display additional selection dialogs.

Solution Explorer

The Solution Explorer window organizes all essential components, and provides access to other dialogs that set additional project options.

The Solution or Project file tracks all the components that make up the final executable file. It also sets the compiler options ranging from whether to include debug code or not, to setting a preferred optimization method. The majority of actions you can execute in the Solution Explorer are found in the Popup Menu options.

If you use the Application Generator to create your source code, the only thing you will probably use the Project System for is to set debugging options.

Tree List

The Tree List itemizes the file level elements which comprise your project, including source code files, file drivers, other projects to compile, external libraries and resources, and other programs to execute as part of the make process.

Note: Assemblies that are referenced by default by Clarion# compiler now visible in the Project Browser (grayed) and available for code completion.

Properties

The Properties button  calls a dialog allowing you to specify compile options for the selected item, or the entire project. The particular dialog which appears depends on the currently selected item. You can also right-click on any tree item, and select the Properties menu item (when appropriate).

When you select the root project item (e.g., the project itself), the **Global Options** dialog appears. This allows you to set compile options for the project.

When you select a source code file, the Properties dialog appears. This provides a limited set of options that apply only to the selected file.

Show All Files

Press the Show All Files button  to show all appropriate project files.

Refresh

Press the Refresh button  to refresh and collapse the project tree.

Open

For hand-coded projects, if a source code file is selected,  calls the Text Editor and loads it into a source code document window.

Right-clicking on selected tree items enable a popup menu with the following items:

Build/Rebuild/Clean Solution

Available on the solution and project nodes, select these items to compile (build) the active project.

Add	Calls a sub-menu that is appropriate for the selected tree item. You can add new or existing projects to the current project, new folders and items, and drivers, libraries, resources, and source files where appropriate. You can specify that another project is built in the course of building the current project. You can add icons that link into your executable so they do not have to be shipped separately. For hand coded applications, you can insert new .CLW files. Database Drivers can also be added here. This section is also used to include manifest files in hand coded projects. You can also include *Version Info Script* files here. These files allow you to enter and control specific version information for this application. A script file must have a *.version* extension. For more information, see Version Information Resource Files. You can also add external resources, such as .CUR, or .ICO files if they were not explicitly named as attributes in your source code. For example, if you specify variable naming a bitmap file in an IMAGE control (such as !MyBMP.BMP), you can link it in by adding it to the Project Tree below this item.
Delete	Allows you to remove the currently selected item from the Project Tree. This does not physically remove the file from disk.

Standard options for Cut, Copy, Paste and Rename are also available via the popup menu.

Solution Explorer Popup Menu Options

Each branch of the Solution Explorer had a set of popup menu items that pertain to that particular area. Many of these items are also available through the IDE Main Menu. Below is a summary of these pop up menu items

The Solution Branch

Build Solution	Builds the active solution. The **Build Solution** option compiles only those project files and components that have changed since the last build.
Rebuild Solution	The **Rebuild Solution** option builds all project files and components irrespective of the changes made to them.
Clean Solution	Deletes all files from the target project folders. This action uses an XML based "FileList" that was generated by the first Build activity. If you attempt to clean a solution that has never been built, you will receive a project warning that it could not be cleaned.
Add > New Project	Used to add a new project to the existing solution. Opens the IDE New Project dialog
Add > Existing Project	Used to add an existing project to the Solution tree. Opens a filtered File Dialog window with valid projects displayed. Valid project may be in current MSBuild format, or legacy project files used in prior version of Clarion.
Add > Item	Used to add any item to the current Solution tree. Opens a generic File Dialog window listing all files.
Add > New Solution Folder	Used to add a new folder to the current solution. Examples of Items stored in this folder might be documentation or other references that are not directly a part of the Build process.
Paste	Used to Paste other Projects to the target Solution
Rename	Used to rename the current Solution. You can make your solution more descriptive if needed.
Properties	Opens the Properties Pad. In the Solution branch, this list is empty.
Edit Project Dependencies	Opens the Project Dependency Editor, which allows you to add and remove project dependencies.

The Project Branch

Every solution must have at least one project branch. In the case of multi-DLL projects, there is usually more than one.

Build/Rebuild/Clean	Similar in function to the Solution options discussed above, but focuses only on the current project selected.

Add > New Item	Opens the **Add File** dialog, where you can add a Program MEMBER, a main PROGRAM or Stand Alone MEMBER file.
Add > Existing Item	Used to add existing files to the target project
Add > File Driver	Opens the **Select File Drivers to include…** dialog, where you can add new File Driver libraries to the active project.
Add > Libraries, Objects and Resources	Opens the **File Dialog** window, where you can add any file, including *Library and Object* Files, *Icon* Files, *Cursor* Files, *Image*, *Manifest*, and *Version Resource* files.
Add > Referenced Projects	Opens the **Add Reference** dialog, used to add additional projects that are referenced by the selected project. This feature is essential for multi-DLL projects to resolve the project references.
Run Project	Runs the active project, if applicable. Target must be an executable.
Debug Project	Runs the active project in Debug mode, if applicable. Target must be an executable. Invokes the Clarion 7 Debugger.
Set as StartUp Project	Sets the project as the one the IDE will execute when you click the Run button. Your solution may have more then one target executable project.
Remove	Removes the selected project from the active solution
Open File using Redirection File	Using the active redirection file (set by the active version), enter a file name to load (open).
Edit Redirection File	Opens the current redirection file that is referenced by the active product version setting.
Create Redirection File in the project directory	Create a new local copy of the version redirection file in the project directory. Commonly known as a local RED file.
Properties	Opens the Project Options dialog

File Drivers

Add File Driver	Opens the **Select File Drivers to include…** dialog
Remove	Used to remove any file driver in the project that is not needed.

Libraries, Objects and Resources

There is only a single option here to Add additional libraries, objects or other resources as needed

Projects to include

There is only a single option here to Add additional projects elements as needed.

Referenced Projects

There is a single option to open the **Add Reference** dialog, in which you can multi select and add multiple projects in one session. All references are automatically resolved by the Project System.

Source File Branch

There are usually multiple source files displayed for each project. For each file displayed, right click for the following popup options.

Open	Opens the selected file into the IDE Source Editor. You can also double-click on the source file to open it.
Add	Displays the same sub-menu as the **Add** menu for the Project Branch. Items added here are added to the entire Project.
Exclude from Project	Excludes the selected file from the Build process.
Rename/Delete	Rename or permanently delete the selected source file
Properties	Open the **Properties Pad** with pertinent project information regarding the selected source file.

Toolbox View

Toolbox View

The Toolbox View provides a list of handy tools that you can use for any target source file. The similarity between all tools is that you can drag and drop a selected item directly to source, or to a target Window/Report Designer where appropriate.

The following tools are currently available always:

ASCII Table

The ASCII Table contains the ASCII Decimal and Hex code, and actual character for each. Only displayable characters are shown.

General

Currently this tool is empty and inactive.

Clipboard Ring

The Clipboard Ring displays the text of recent items copied to the clipboard (CTRL + C).You can quickly Paste these items to a selected source area by dragging the selection from the Toolbox.

When the Designers are active, the following tools are also available:

Clarion Window Controls (Clarion 7 Only)

Displays a list of valid controls used in any WINDOW structure. Click and drag the selected control and drop in the work area of the target window structure while the Designer is active.

Clarion Report Controls

Displays a list of valid controls used in any REPORT structure. Click and drag the selected control and drop in the work area of the target report structure while the Designer is active.

The family of Clarion# Designers display additional categories of controls. To add or modify new categories to the toolbox, right-click on any area of the toolbox and select the **Configure Sidebar** option.

Configure Sidebar

The **Configure Sidebar** dialog displays special add on control assemblies that allow you to extend your available library of controls.

Adding a new category is a two step process:

1. 1. Press the **New** button to add a new **Category**.
2. 2. Highlight the category just added, and press the **Add Components** button. This will pop up an additional dialog that will allow you to select a valid DLL assembly from the Global Assembly Cache (GAC), or from another location (**Custom**)

Add Components

This dialog is used with the Configure Sidebar interface, and displays when you add any components to an existing category.

It allows you to select a valid DLL assembly from the Global Assembly Cache (GAC), or from another location (**Custom**).

The IDE is intelligent and detects that the assembly you have selected is valid or not for the category.

Performance Tips (IDE)

Performance Tips

This topic contains tips to improve the overall performance of the Clarion IDE on your computer, and the minimum and recommended system requirements.

General tips:

1. Turn off virus checking on the Clarion.exe program.
2. Defragment your hard disk after you install Clarion.
3. Check that your system meets the RAM and processor recommendations and then consider whether to upgrade the memory or disk speed.

System Requirements:

Minimum

1.6 GHz CPU, 384 MB RAM, 1024x768 display, 5400 RPM hard drive

Recommended:

2.2 GHZ or higher CPU, 1024 MB or more RAM, 1280x1024 display, 7200 RPM or

higher hard drive. If Running on Windows Vista: 2.4 GHz CPU, 768 MB RAM

IDE Tips

When you have the Clarion integrated development environment (IDE) open:

1. Before you exit the IDE, close the pads that you do not use at startup. This can increase your startup speed the next time that you start the IDE.

2. Do not display the Properties Pad at startup. The Properties Pad displays automatically when you open a solution.

Processor and Memory

For the best performance of the Clarion IDE, you want to get a dual core or better CPU. We also recommend getting at least 2GB or more of RAM. Make sure you always get the fastest possible hard-drive when buying a new machine - and where necessary trade off purchasing additional CPU processor speed in favor of a faster disk instead.

Hard Drives

The default hard drive speed for most PC laptops is typically 5400rpm - which is a pretty slow drive in comparison. If you are getting a new laptop and plan to use Clarion on it, we recommend making sure you get a 7200rpm drive instead. You will realize a significant performance benefit by doing this.

For a desktop machine hard drive, consider getting a 10000rpm hard drive, or even better a 15000 RPM drive. These make a very big difference over the default 7200rpm drives that typically come with desktops. You should also check the drive's cache size. Older drives have 1MB or 2MB of cache. But you can get drives with 8MB and 16MB caches which will speed up the performance of the drive.

You could also consider buying a second physical hard drive and setup your operating system and OS virtual memory swap file to use one of the physical drives, and store all your data on the second physical drive. The benefit is that your read/write data operations won't be competing for disk I/O activity with the operating system updating the virtual memory file.

Project System

Database Driver Libraries

Your application calls various database driver routines to access your database tables. These routines are in libraries supplied with Clarion and installed by default in the \LIB subdirectory by the Clarion setup program. During the *link* process, references to these external routines can only be resolved if the library containing the routines is added to your project file.

> The Application Generator automatically adds the appropriate driver libraries based on the Data Dictionary table driver selections and the Project System settings. For hand-coded projects, you should manually add the appropriate driver libraries.

To link a database driver library, **highlight** *Database driver libraries* in the Project Tree list. Then **press** the **Add File** button and select the driver to add from the **Select Driver** dialog.

Generalize Name

The Generalize Name option is found by right-clicking on any library name in the project tree. The IDE reads the library and attempts to generalize key designations, like link mode and version.

For example, given the following library name:

C60PRLBX.LIB

The Transformed Library Name will become:

C%V%PRLB%X%.LIB

This allows you to easily switch versions without having to remove and include new libraries.

New Project File Dialog

This dialog allows you to type in the basic information the Project System needs to create a new project file (.CWPRJ for Win32, .CNPRJ for Clarion.NET) and solution file (*.SLN).

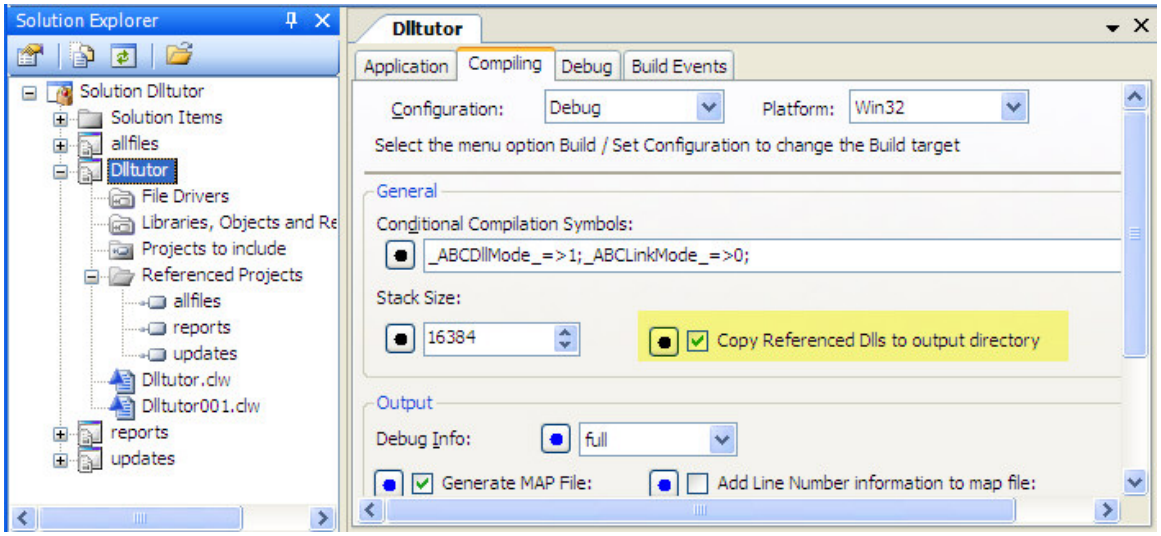
Categories	Select a valid category. For Win32 projects, select *Clarion for Windows*. For Clarion.NET, you have three sub categories to create desktop (*Window Applications*), web based (*ASP.NET*), or Mobile (*Compact Framework*) projects.
Quick Starts	Specify the type of project that you wish to create. In Clarion for Windows, the Quick Starts available are hand coded DLL, EXE, or Library projects, or *Applications* (from scratch or from an existing TXA file). A TXA file is an application in a special text format. Select the Application Quick Start here. Clarion.Net choices are more diverse, depending on the sub-category you havew selected.
Name	Type in a project name which displays at the top of the Tree List in the Project View.
Location	Type in (or select with the Browse for Folder dialog after pressing the ellipsis button) the location where the project will be created.
Create directory for Sources	Check this box to specify a subdirectory with the project name that will be created at the Location you specified.
Auto create project subdir	Check this box to autocreate a project folder that uses the project Name and Location specidfied. For example, if the name of the project is *Orders* and the Location is *C:\Example,* a sub-folder named *Orders* will be created in the *Example* folder.

Project Copy System

The new project system copy system is very complete and flexible.

Whenever you build an executable (EXE), all necessary dynamic link libraries (DLLs) will be copied to the directory where that EXE is located.

In the following list the term "valid referenced project" means any referenced project that does *not* have the **Copy Child Dlls to output directory** property set to *Never* (which is visible in the Properties pad when you highlight any referenced project) and has the **Copy Referenced Dlls to output** property set to TRUE (which is visible on the **Compiling** tab of the Project Property window).



The system finds the DLLs based on:

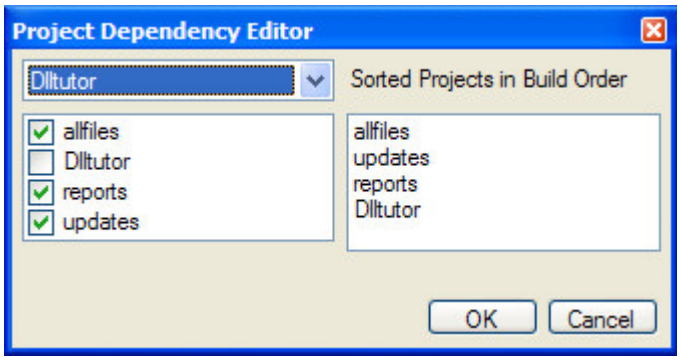
- Referenced projects that DO NOT have **Copy Dll to output directory** set to *Never*.
- Library (LIB) files in this project that DO NOT have the **Copy Dll to output directory** set to *Never*
- Library (LIB) files in a valid referenced project that does not have the **Copy Dll to output directory** property set to *Never*.
- File drivers included in this project
- File drivers included in a valid referenced project
- Library (LIB) files included via PRAGMA statements in the code for this project
- Library (LIB) files included via PRAGMA statements in the code for a valid referenced project.
- Files in the project with the **Copy to output directory** property not set to *Never*
- Files with the **Copy to output directory** property not set to *Never* in a valid referenced project
- The active runtime DLL (c%V%run.dll)

Overriding all of this is the *LibIgnoreList.xml* file located in the *Application Data\SoftVelocity\Clarion\7.0* directory. Any library file in that file will never have its 'matching' DLL copied. This file is to be used for library files that do not have a matching DLL.

Note that the copying only happens when the target of the project is an EXE and only if you have the **Copy Referenced Dlls to output** property set to TRUE (the default).

Project Dependency Editor

The **Project Dependency Editor** is called from the popup menu of the Solution Node in the Solution Explorer.



The list on the left displays dependencies of the selected project in the drop list above. Check a project to easily add it as a dependency, or uncheck to remove it.

On the right, the build order of all project is displayed. Changing the dependencies in the left list will automatically change the order on the right when needed.

Project Redirection

A new feature of the C7 project system is the ability to set default project settings through redirection.

solglobals.pi and cwglobals.pi

Whenever a project is built with C7, the project system uses the redirection file to look for a file called *solglobals.pi*. If it finds it, this file is included at the beginning of the project. Regardless of the existence of *solglobals.pi,* it then looks for a file called *cwglobals.pi*. If this is found, it is then included into the project. Finally, the project file is read.

This process allows you to put project settings specific for a solution in *solglobals.pi* and project settings you want to set for all your projects in *cwglobals.pi*.

Project Redirection Tasks

The C7 project system uses MSBuild to build projects and solutions. MSBuild uses XML files to control how projects are built. You should look in the Microsoft help for detailed information on how MSBuild files work.

One feature of MSBuild is that you can add extra tasks. A currently undocumented task available to Clarion developers is the Redirection task. This allows you to get the full path of a file using the redirection system.

To use this task in an MSBuild project you need to add the following line to the top of the file:

```
<UsingTask TaskName="Microsoft.Build.Tasks.Redirection"
AssemblyFile="$(ClarionBinPath)\SoftVelocity.CW.Build.Tasks.dll"/>
```

The following table describes the parameters of the Redirection task.

Parameter	Description
File	Required String parameter. Specifies the file that should be found via redirection
Version	Optional String parameter. Specifies the version of Clarion to be used. If not specified, the current version of Clarion is used
ForOpen	Optional Boolean parameter. If true, OutputPath is set to the path where the file exists. If false, OutputPath is set to the path where the file will be created. If not specified, true is assumed. If ForOpen is true and the file does not exist, OutputPath is set to an empty string.
RedirectionDirectory	Optional String parameter. The directory where the redirection file is located. If not specified, the current directory is used.
OutputPath	Optional String output parameter. Contains the full path of the file.

Example:

The following code example creates a new item collection named *MySourceItemsWithMetadata* from the item collection *MySourceItems*. The *CreateItem* task populates the new item collection with the items in the *MySourceItems* item collection that contain *MyAddMetadata* values of true. It then adds an additional metadata entry named *MyMetadata* with a value of Hello to each item in the new collection.

After the task is executed, the *MySourceItemsWithMetadata* item collection contains the items *file1.resx* and *file3.resx*, both with metadata entries for *MyAddMetadata* and *MyMetadata*. The *MySourceItems* item collection is unchanged.

```
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

 <UsingTask TaskName="Microsoft.Build.Tasks.Redirection" AssemblyFile="$(ClarionBinPath)\SoftVelocity.CW.Bu


  <Target Name="FindFile">

   <Redirection

    File="cwglobals.pi"

    ForOpen=true

    Version="Clarion 6.0 Enterprise Edition"

    RedirectionDirectory="c:\mydatadir">

    <Output

     TaskParameter="OutputPath"

     PropertyName="FullPath"/>

     </Redirection>

     <Message Text = "File does not exist"

      Condition="$(FullPath)=""/>

     <Message Text = "Full Path is $(FullPath)"

      Condition="$(FullPath)!=""/>

  </Target>

</Project>
```

Project System Quick Start

The Clarion 7 project system has been updated and upgraded to use the MSBuild project engine. MSBuild is the new extensible, XML-based build engine that ships with the .NET Framework 2.0 and higher. MSBuild simplifies the process of creating and maintaining build scripts, and formalizes the steps required to institute a formal build process.

The complete reference guide to MSBuild can be found at the following link:

http://msdn2.microsoft.com/en-us/library/wea2sca5(VS.80).aspx

All Clarion 7 projects use a default extension of *.CWPROJ.

Here is a sample listing:

```
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003"
>

  <PropertyGroup>

    <OutputName>people</OutputName>

    <Model>Dll</Model>

    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>

    <Platform Condition=" '$(Platform)' == '' ">Win32</Platform>

    <DefineConstants>_ABCDllMode_=&gt;0;_ABCLinkMode_=&gt;1</DefineConstants>

    <ProjectGuid>{FFFF1465-30CB-4F2D-A01C-AF2E646CA106}</ProjectGuid>

  </PropertyGroup>

  <PropertyGroup Condition=" '$(Configuration)' == 'Debug' ">

    <vid>full</vid>

    <check_stack>True</check_stack>

    <check_index>True</check_index>

  </PropertyGroup>

  <PropertyGroup Condition=" '$(Configuration)' == 'Release' ">

    <vid>off</vid>

    <check_stack>False</check_stack>

    <check_index>False</check_index>

  </PropertyGroup>

  <ItemGroup>
```

```
    <FileDriver Include="TOPSPEED" />

    <Compile Include="PEOPLBC0.CLW" />

    <Compile Include="PEOPLBC.CLW" />

    <Compile Include="people.clw" />

    <Compile Include="peopl001.clw" />

  </ItemGroup>

  <Import Project="$(ClarionBinPath)\SoftVelocity.Build.Clarion.targets" />

</Project>
```

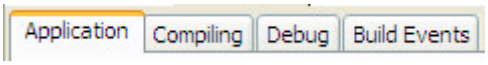
All projects compiled in Clarion 7 are stored in solution files (*.SLN). These files can store multiple projects, and allow multiple resources to be referenced and included.

Think of the projects used in earlier versions of Clarion as a subset of any solution.

For your convenience, we also have included the project system documentation for the versions of Clarion prior to Version 7.

Project Properties - Clarion 7

Project Properties



These dialogs allow you to set compile options for the project, or its components. The options vary according to the item selected in the Project Tree dialog at the time you press the **Properties** button.

When you select a "folder level" item (such the project itself), the **Project Properties** dialog appears. This allows you to set compile options for the project.

Throughout the project settings you will notice the following button:

This button allows you to choose in selected areas how a project setting will be stored. *Configuration-specific* means the setting can have a different value in each configuration (Debug/Release). *Platform-specific* means the setting can have a different value on each platform (target O/S). You can also choose to store the setting in the *.user-file* instead of the project file, so that every developer can use a different value without getting conflicts in source control systems.

By default, all setting are project specific.

The **Project Properties** dialog contain the following options.

Configuration Specify which Project configuration options you wish to set. Select *Build* or *Release* from the drop list provided. To change the Build target, use the **Build/Set Configuration** menu option.

Platform Current default for Clarion 7 is *Win32*.

Application

Configuration Specify which Project configuration options you wish to set. Select *Build* or *Release* from the drop list provided. To change the Build target, use the **Build/Set Configuration** menu option.

Platform Current default is *AnyCPU*.

Target Name The output filename of the project. The Project System will use the description next to the Project name in the Project View tree list. It will also be the default project target filename (<*output name.output.type*>). Macros are allowed, but only %V%, %X% and %L%.

Output Path The default target is based on the Project folder name and Build Configuration (e.g., <*projectfolder*>*debug* or <*projectfolder*>*release*). Use this option to specify an alternative target path where the target EXE, LIB, or DLL will be copied to. The only time you might want to use this is if you want to temporarily output a program you are creating to a different directory to preserve an old executable.

Output Type Specify the type of executable file: choose **.EXE, .LIB**, or **.DLL** from the **Target Type** drop down list.

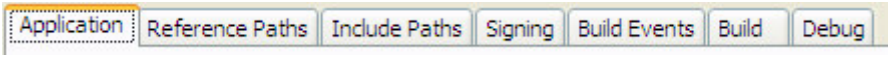
Link Mode Specifies how the runtime library is called by the target file: choose ***DLL LIB***, or ***CustomDLL*** from the **Link Mode** drop down list.:

DLL	Uses the Clarion runtime library DLL (and database driver(s) .DLLs). It is called **C70RUN.DLL**.
LIB	Links the runtime library and any database drivers into your executable using Smart Method Linking (only the necessary portions are linked in). This creates a "one-piece" executable.
CustomDLL	Specifies that another External DLL contains the runtime libraries and database drivers. *CustomDll* means that all of the Clarion runtime DLLs have been built into a DLL that the user (you) has provided. As such no file drivers can be linked into the project. *CustomDLL* is essentially useless for template driven applications unless its your own custom template chain that doesn't add any of the Clarion drivers. The *CustomDll* setting is best suited for hand coders.The calls to this DLL must be exported.

Application icon By inserting .ICO files after the Application icon item, you can link the icons into your executable so they do not have to be shipped separately.

Project Information Displays the **Project folder**, **Project file** and **Output name**. These items are read-only.

Build Events



This dialog is used to specify optional commands to execute before and after the actual build sequence. Build events are executed only if the build successfully reaches those points in the build process.

Build Events are useful for modifying manifest information, or updating build version numbers.

Compiling

Conditional Compilation Symbols

To define a switch, or switches, for use with the COMPILE and OMIT compiler directives, type a list of valid Clarion labels separated by semicolons. Each label defines a separate switch.

For example, type 'Demo' in the entry field. The Project System will create a switch called Demo and turn it "on." Now you can use the switch in conditional COMPILE and OMIT statements within your source code. For example:

```
COMPILE('END COMPILE',DEMO=ON)
 IF TODAY() > FirstRunDate + 30
  ReturnCode = MESSAGE('Beta period expired')
  RETURN
 END
END COMPILE
```

You can also add new defines in the Conditional Compilation Symbols Dialog by pressing the ellipsis button to the right of the entry prompt.

Stack Size

To specify the stack size, type a value in Kilobytes in the **Stack Size** field.

Copy Referenced DLLs to Output Directory

This will copy the core dynamic linked libraries DLLs to the directory where the executable (or DLL) is being created. This means you no longer need to have different versions of Clarion on your environment path, thus allowing you to easily test with multiple versions of the same product. **The redirection (RED) file is used to find DLLs when this check box is active.**

The PATH of core DLLs is automatically added by the installer. The same is of course true for all earlier versions of Clarion. So unless you have the case where you are maintaining separate distinct versions of C7 (or C6 etc.) then you do not need to use this option.

The Project system uses the RED file to locate the DLLs. The standard RED file we ship has this entry in the *Copy* section;

```
[Copy]
-- Directories only used when copying dlls
*.dll = %BIN%;%ROOT%\Accessory\bin
```

Also. settings in a RED file found in the project directory take precedence over the base RED file settings.

Examples of files copied would be file driver DLLs, runtime library DLLs, and any other DLLs that you have referenced in the active project.

Debug Info

Specifies the level of debug capability, choose ***Off***, ***Min***, or ***Full*** from the drop down list.

Generate MAP File

Creates a map file, which contains information about segment sizes and public functions. The map file may be used with third party debuggers.

Add Line Number information to map file

Check this box to build line numbers into the object file. This is not necessary for the Clarion debugger, but may be helpful when using other debuggers.

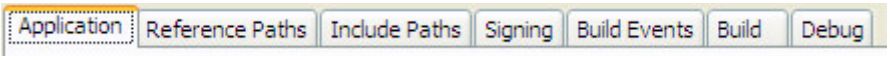
Index out of range

Enables "array index larger than the array size" warnings at runtime.

Stack Overflow

Enables stack overflow warnings at runtime.

Debug



The options on this tab control actions when the debugger is called within the IDE. You can control the *Start Action* and other *Start Options*.

Start Action The debugger can be launched in one of three ways. **Start Project** loads the debugger with the current active project. **Start external program** will launch an external program specified when the debugger is launched, or select the **Start browser in URL** so you can start a specific web page when the debugger is started.

Start Options Optionally specify **Command line arguments** that will be used for the specific start action selected. You can also specify a **Working directory** as the debugger is launched.

Report Designer

Break Properties

This dialog lets you add or edit the properties of a group break.

Design

Locked "Freezes" the break control so that subsequent data dictionary changes are not applied during application synchronization.

Extra

IgnoreCase Check this option if you would like to ensure a case insensitive break logic. This option is useful when a report is sorted by a case insensitive key, which could cause a "false break" between two string values that only vary in case (i.e., Defranco and DeFranco).

General

Label Type a valid Clarion labelDeclaration_and_Statement_Labels, naming the BREAKBREAK__declare_group_break_structure_ structure.

Use This defines the USE attribute for the BREAK structure. Type a field equate label to reference the BREAK structure in executable code.

Variable Type a variable name, to generate a break when the value changes as you sequentially process the file.

Group breaks provide a means of breaking the data into sections and optionally providing subtotals. Each group gathers a set of data file records, all sharing the same value in the BREAK field. Within a report, you may visually separate these rows, and add a subtotal or summary information, usually below the group. Group breaks are also called group bands by some popular end user database applications.

The group break may contain the same elements as the report: a group HEADER, group DETAIL, and group FOOTER. These structures all print inside the DETAIL print area, each following the other by any offset specified in their AT attributes.

Though they print on the page at the same time, the application composes the group HEADER before the group DETAIL. The group HEADER is a good place to identify the group.

The group FOOTER, is composed after the group DETAIL. You can place a string saying "Total:" followed by a string variable which contains the field to be summed, with the SUM attribute.

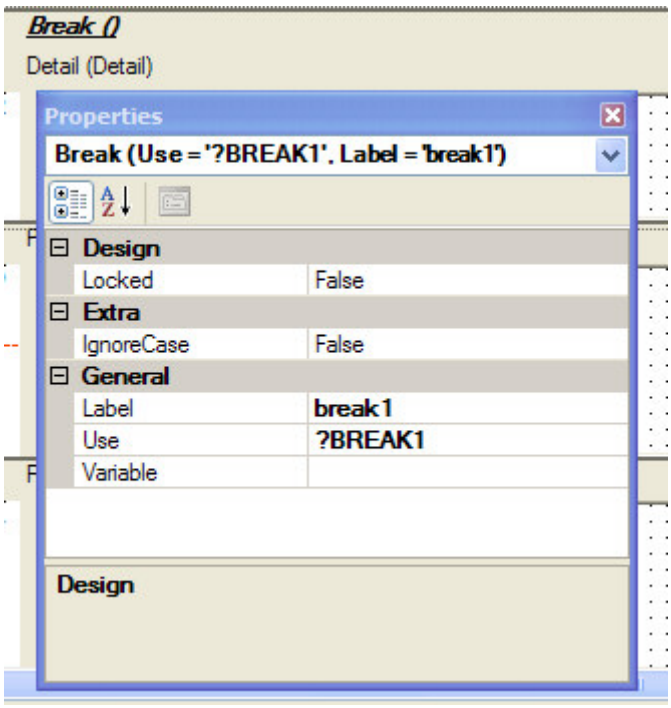
To create a group break:

1. Click on the DETAIL band where you wish to add the break.

2. From the IDE Menu, choose **Report Designer** ▸**Bands** ▸**Surrounding Break**.

This inserts the group BREAK.

3. Click on the ***Break( )*** just created. If the Property View is not visible, open it now from the IDE Menu (**View ▸Properties)**



4. In the **Break Properties** dialog, type the name of a variable or field, including the prefix, to break on in the **Variable** property.

5. Type a valid Clarion label in the **Label** property to name the break.

When the report prints, it groups all records with the same value for the BREAK field, as well as the group HEADER and FOOTER.

Tip

If the break variable is a global or local variable, you must be sure that the executable code updates its value, so that it can generate a group BREAK.

See Also:

How to Set Report Group Breaks

Quick Links

Embeds Accesses the **Embedded Source** dialog for this procedure.

Break Group Header Properties

This dialog lets you edit the properties of the group HEADER. To specify an element to compose at the start of each group, place it in the group HEADER.

Though they print on the page at the same time, the application composes the group HEADER before the group DETAIL. The group HEADER is a good place to identify the group, for example, with a label saying "Customer:" followed by a variable string for the customer name field.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Freeze — "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Extra

Absolute — Specifies that the HEADER section always prints at the same fixed position on the page. This adds the ABSOLUTE attribute to the DETAIL structure.

Alone — Specifies that the HEADER section always prints alone on a page.

Together — Specifies that the band is to only print on a single page, rather than being automatically split if there is not sufficient room for it. This adds the TOGETHER attribute.

General

Layout — Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

Page After To print the HEADER, then force a new page, type a value in the **Page after** box. This sets the PAGEAFTER attribute. This prints the HEADER, then begins a new page.

The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page after** field.

Page Before To print the HEADER structure on a new page, type a value in the **Page Before** box in the **Detail Properties** dialog. This sets the PAGEBEFORE attribute. The print engine generates a page break before printing at the top of the next page.

The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page Before** field.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the HEADER in executable code.

With Next To prevent 'widow' elements in a printout, type a value in the **With Next** field. This sets the WITHNEXT attribute. A 'widowed' print element is one which prints, but then is separated from the succeeding elements by a page break.

The value specifies the number of succeeding elements to print--a value of '1,' for examples, specifies that the next element must print on the same page, else page overflow puts them on the next.

With Prior To prevent 'orphan' elements in a printout, type a value in the **With Prior** field. This sets the WITHPRIOR attribute. An 'orphaned' print element is one which prints on a following page, separated from its related items.

The value specifies the number of preceding elements to print--a value of "1," for example, specifies that the previous element must print on the same page.

Position

Lets you set the location and size of the group header. The position is relative to the top left corner of the paper.

The **Position** dialog lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position visually, by dragging with the mouse in the **Report Designer's Page Layout View**.

To set a location or size for your report section, specify fixed coordinates with this dialog. Choose from the following options for the **Top Left Corner**, the **Width**, and the **Height**.

Default Sets a value based on the Paper Size and Print Detail position (see also **Report Properties** dialog).

Value To set a specific position and size, mark the **Fixed** choices and enter a value in the entry box.

The measurement units for these boxes are specified on the **General** tab of the **Report Properties** dialog.

See Also:

How to Set Report Group Breaks

Quick Links

Embeds Accesses the **Embedded Source** dialog for this procedure.

Break Group Footer Properties

This dialog lets you edit the properties of the group FOOTER.

The group FOOTER is composed immediately after the group DETAIL, and provides the logical place for adding subtotals to your report.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

General

Layout Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

Page After To print the FOOTER, then force a new page, type a value in the **Page after** box. This sets the PAGEAFTER attribute. This prints the FOOTER, then begins a new page.

The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page after** field.

Page Before To print the FOOTER structure on a new page, type a value in the **Page Before** box. This sets the PAGEBEFORE attribute. The print engine generates a page break before printing at the top of the next page.

The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page Before** field.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the FOOTER in executable code.

With Next To prevent 'widow' elements in a printout, type a value in the **With Next** field. This sets the WITHNEXT attribute. A 'widowed' print element is one which prints, but then is separated from the succeeding elements by a page break.

The value specifies the number of succeeding elements to print--a value of '1,' for examples, specifies that the next element must print on the same page, else page overflow puts them on the next.

With Prior To prevent 'orphan' elements in a printout, type a value in the **With Prior** field. This sets the WITHPRIOR attribute. An 'orphaned' print element is one which prints on a following page, separated from its related items.

The value specifies the number of preceding elements to print--a value of "1," for example, specifies that the previous element must print on the same page.

Extra

Absolute Specifies that the FOOTER section always prints at the same fixed position on the page. This adds the ABSOLUTE attribute to the FOOTER structure.

Alone Specifies that the FOOTER section always prints alone on a page.

Together Specifies that the band is to only print on a single page, rather than being automatically split if there is not sufficient room for it. This adds the TOGETHER attribute.

Position

Lets you set the location and size of the group footer. The position is relative to the top left corner of the paper.

The **Position** dialog lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position visually, by dragging with the mouse in the **Report Designer's Page Layout View**.

To set a location or size for your report section, specify fixed coordinates with this dialog. Choose from the following options for the **Top Left Corner**, the **Width**, and the **Height**.

Default Sets a value based on the **Paper** property size and Print Detail position (see also **Report Properties** dialog).

Value To set a specific position and size, select the appropriate **Value** choices and enter a value in the entry box.

The measurement units for these fields are specified in the **General** category of the **Report Properties** dialog.

See Also:

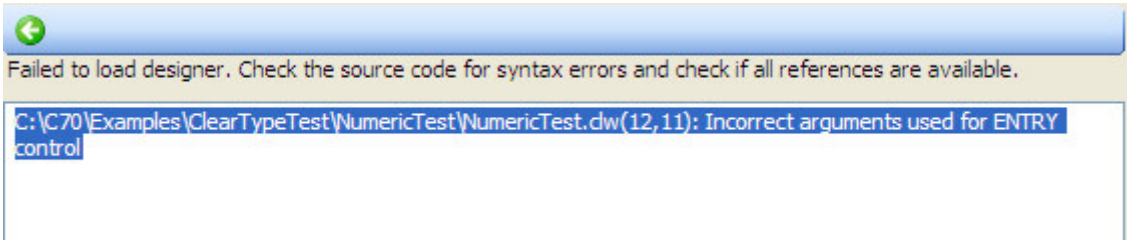
How to Set Report Group Breaks

Quick Links

Embeds	Accesses the **Embedded Source** dialog for this procedure.

Designer Error Window

The Designer Error window is a special window that provides specific information in the event that a window or report structure could not be loaded for any reason.



In the Clarion 7 Designers, the most likely reason is a missing or incorrect element contained within the referenced structure.

In the Clarion# designers, the most likely cause is a missing component or declaration in the Clarion# project source.

Usually, the information presented will refer you to a precise line in the source from where the problem originated.

Detail Band Properties

This dialog lets you edit the properties of the report detail.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Report Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

General

Label Type a valid Clarion labelDeclaration_and_Statement_Labels, naming the DETAILDETAIL__report_detail_line_structure_ structure.

Layout Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

Page After To print the DETAIL, then force a new page, type a value in the **Page after** box in the **Detail Properties** dialog. This sets the PAGEAFTER attribute. This prints the DETAIL, then prints the page FOOTER, then begins a new page.

To print a separate page for each record, place the variable strings and/or controls you wish in the DETAIL, and specify the PAGEAFTER attribute in the Detail Properties dialog.

The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page after** field in the **Detail Properties** dialog.

Page Before To print the DETAIL structure on a new page, type a value in the **Page Before** box in the **Detail Properties** dialog. This sets the PAGEBEFORE attribute. The report prints the full DETAIL starting at the top of the next page. The report FOOTER, however, prints on the first page. The page number automatically increments, unless you reset it. To reset the page number to a value you specify, type it in the **Page Before** field.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the Detail in executable code.

With Next To prevent 'widow' elements in a printout, type a value in the **With Next** field. This sets the WITHNEXT attribute. A 'widowed' print element is one which prints, but then is separated from the succeeding elements by a page break. The value specifies the number of succeeding elements to print--a value of '1,' for examples, specifies that the next element must print on the same page, else page overflow puts them on the next.

With Prior To prevent 'orphan' elements in a printout, type a value in the **With Prior** field. This sets the WITHPRIOR attribute. An 'orphaned' print element is one which prints on a following page, separated from its related items. The value specifies the number of preceding elements to print--a value of "1," for example, specifies that the previous element must print on the same page.

When placing subtotals or totals in a DETAIL, use the WITHPRIOR attribute to insure they print with at least one member of the column above it when a page break occurs.

Extra

Absolute Specifies that the DETAIL section always print at the same fixed position on the page. This adds the ABSOLUTE attribute to the DETAIL structure. Otherwise, the DETAIL prints at a position relative to the last section printed in the detail print area.

Alone Specifies the print engine should print *only* the DETAIL section, without FORM, HEADER, and FOOTER sections. This setting is most useful for report title and grand totals pages. This adds the ALONE attribute to the DETAIL structure.

Together Specifies that the band is to only print on a single page, rather than being automatically split if there is not sufficient room for it. This adds the TOGETHER attribute.

Position

Lets you set the location and size of the print detail, by filling in the AT attribute. The measurement units for these boxes are specified on the **General** tab of the **Report Properties** dialog.

To set a precise starting point for your detail relative to the last detail printed, specify **Top Left Corner** coordinates with this dialog. These settings may also be accomplished visually by dragging the report section and it's borders in **the Report Designer's Page Layout View**.

To set the size of the detail, choose from the following options for the **Width** and **Height**.

Default Specifies to use the default width of the REPORT structure, and the height as specified by the DETAIL contents.

Value To set a specific size for width or height, enter the specific **Value** choices.

See Also:

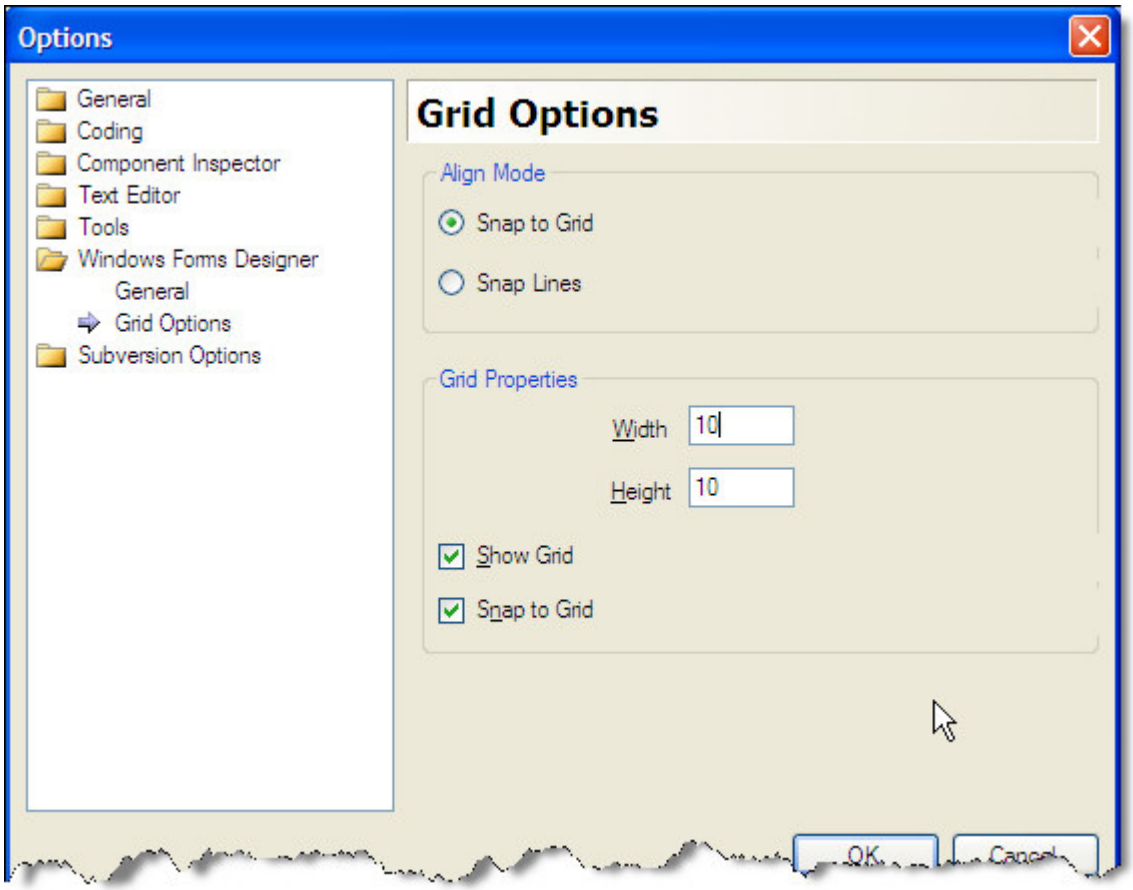
How the Print Engine Processes Report Sections at Runtime

Quick Links

Embeds Accesses the **Embedded Source** dialog for this procedure.

Grid Options

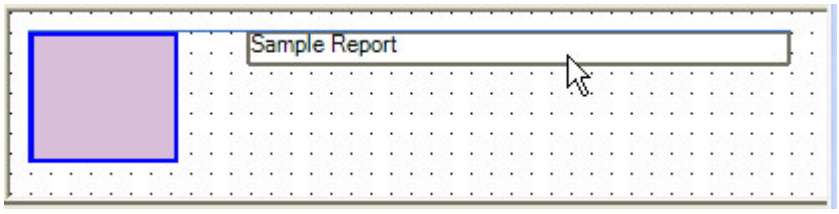
Grid Options are located in the IDE menu by selecting the **Tools ▸Options** menu item. In the *Options* dialog, slect the *Forms Designer* option, and the *Grid Options* setting as shown below:



This allows you to set the spacing between grid points. When you place a new control in the report, you can then optionally force it to the grid points with the **Report Designer ▸Format ▸Align ▸To Grid** command. You can also simply right-click on the control and select the **Align to Grid** popup menu item.

Set the spacing by typing in values for the **Width** and **Height** spacing. The measurement units depend on the default for the report, as set in the **Report Properties** dialog. The *Grid Options* dialog tells you the minimum value (1/20th inch, or 2 millimeters).

The **Snap Lines** option places the control in alignment with other controls using line indicators as shown below:



Page Header Properties

This dialog lets you edit the properties of the page HEADER. To specify an element to compose at the start of each report page, place it in the page HEADER.

Though they print on the page at the same time, the print engine composes the page HEADER before the page DETAIL, or FOOTER. The page HEADER is a good place for company logos, print dates, etc.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Report Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Extra

Together Specifies that the band is to only print on a single page, rather than being automatically split if there is not sufficient room for it. This adds the TOGETHER attribute.

General

Layout Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the page HEADER in executable code.

Position

Lets you set the location and size of the page header. The position is relative to the top left corner of the paper.

The **Position** dialog lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position visually, by dragging with the mouse in the **Report Designer**.

To set a location or size for your report section, specify fixed coordinates with this dialog. Choose from the following options for the **Top Left Corner**, the **Width**, and the **Height**.

Default	Sets a value based on the Report settings.
Value	To set a specific position and size, mark the **Fixed** choices and enter a value in the entry box.

The measurement units for these boxes are specified in the **Report Properties** dialog.

See Also:

How to Control Page Breaks

Using the Report Designer - An OverviewHow_to_Use_the_Report_Formatter___An_Overview

Quick Links

Embeds	Accesses the **Embedded Source** dialog for this procedure.

Page Footer Properties

This dialog lets you edit the properties of the page FOOTER. To specify an element to compose at the end of each report page, place it in the page FOOTER.

Though they print on the page at the same time, the print engine composes the page HEADER before the page DETAIL, or FOOTER. The page FOOTER is a good place for page numbers, page totals, etc.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Report Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Extra

Together Specifies that the band is to only print on a single page, rather than being automatically split if there is not sufficient room for it. This adds the TOGETHER attribute.

General

Layout Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the page FOOTER in executable code.

Position

Lets you set the location and size of the page footer. The position is relative to the top left corner of the paper.

The **Position** dialog lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position visually, by dragging with the mouse in the **Report Designer**.

To set a location or size for your report section, specify fixed coordinates with this dialog. Choose from the following options for the **Top Left Corner**, the **Width**, and the **Height**.

Default	Sets a value based on the Report settings.
Value	To set a specific position and size, mark the **Fixed** choices and enter a value in the entry box.

See Also:

How to Control Page Breaks

Using the Report Designer - An OverviewHow_to_Use_the_Report_Formatter___An_Overview

Quick Links

Embeds	Accesses the **Embedded Source** dialog for this procedure.

Report Form Properties

To specify a static page element which prints on every page, place it in the FORM. This is a free-floating section which can overlap the other sections.

Use the FORM as a layer, to 'hold' graphic frames or preprinted *forms* into which the data from the other sections 'fit.' Another use for the FORM is to hold a 'watermark,' which prints underneath the report.

The FORM size defaults to the same size as the page, less the margins.

The print engine composes the FORM at the beginning of the print job; and does not update it thereafter. Therefore, the FORM is not suitable for holding variable data fields, or even a page number. You can, however, print fields from a control file, if you wish to print the same field contents on every page of the report.

For best results when using a drawing tool to create a 'watermark,' on, for example, a 300 DPI printer, set the fill for the watermark element to 10% gray, or light gray. At higher printing resolutions, try 20% gray.

The FORM should guide the user to the data. You might use lines and boxes, for example, to divide the DETAIL into 'compartments,' grouping data and columns for the user. You may even create a 'greenbar paper' effect by alternating gray or light green color blocks.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Report Designer** adds the COLOR attribute to your header declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked "Freezes" all the controls on the band so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

General

Layout Indicates the orientation of report controls.

Default maintains the current layout setting as defined by the main REPORT structure

Left to Right maintains the original layout specified in the Report Formatter.

Right to Left essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Formatter.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for all the controls in the report section. You may override the section font by setting a different font in the Properties dialog for any specific control. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the structure. Type a field equate label to reference the FORM in executable code.

Position

Lets you set the location and size of the Report Form. The position is relative to the top left corner of the paper.

The **Position** dialog lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position visually, by dragging with the mouse in the **Report Designer's Page Layout View**.

To set a location for the **Top Left Corner** of your form, specify **X** (horizontal) and **Y** (vertical) coordinates with this dialog. **X** = top margin and **Y** = left margin.

To set a size for your Report Form, choose from the following options for the **Width** and **Height**.

Default	Sets a value based on the Report Width and Height.
Value	To set a specific size, select the **Value** choices and enter a value in the entry box.

The measurement units for these boxes are specified on the **General** tab of the **Report Properties** dialog.

See Also:

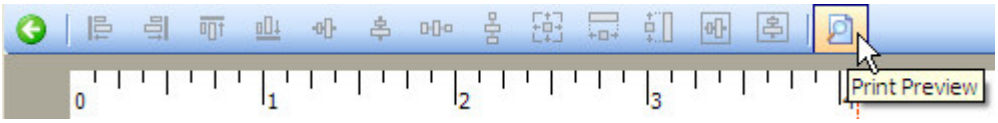
How the Print Engine Processes Report Sections at Runtime

Quick Links

Embeds	Accesses the **Embedded Source** dialog for this procedure.

Preview Print Details Dialog

To preview a report in the Report Designer, press the **Print Preview** button from the Designer toolbar as shown below:



This allows you to choose a report section to preview.

Because you can nest many sections of various types within a single report, you have to select a section before actually previewing it. This way, the Report Designer knows what parts you want to compose on the screen.

Select a section from the **Details** list, then press the **Add** button to move it to the **Selected Details** list. Press the **OK** button to preview.

Report Designer Menu Commands

The **Report Designer** helps you visually design report elements--variable strings and other controls--on screen. The **Report Designer** automatically generates and places the language structures and source code that describe these elements in your .APP file or source code document.

The Report Designer menu is visible during any Designer session, and provides populating controls options and alignment commands for spacing and sizing the controls within the window. You may place two or more controls so that their 'edges' match up with each other. You may also spread the controls out, or make all of them the same size.

To do so, first select two or more controls. Select the first by clicking on it. Select the second and subsequent controls by pressing the CTRL key, then clicking on the second control while the shift key remains pressed.

Lasso multiple controls by CTRL+CLICK+DRAGGING to form a box around the controls.

Duplicate a control by first selecting it, and then press the F2 key or CTRL + DRAG.

Popup (right-click) Menu

The popup menu provides quick and easy access to a subset of the following menu commands.

View Code	Closes the **Report Designer** with option to save (if necessary) and selects the REPORT structure.
Bring to Front	For overlapping controls, select this item to move the active control to the top.
Send to Back	For overlapping controls, select this item to move the active control to the back.
Align to Grid	Select this item to snap the selected control to the nearest grid point
Lock Controls	This option locks all controls to their current position. A "locked" icon appears in the upper left area of the control. When a control is locked, it may not be moved or resized.
Cut	Deletes the selected control from the work area and places it in the clipboard.
Copy	Places a copy of the selected control from the work area into the clipboard.
Paste	Pastes a control from the clipboard into the work area, at the cursor position.
Delete	Deletes the selected control. Alternatively, press the DELETE key.
Properties	Opens the **Properties** pad for the selected control or report band.

Edit Menu

In the IDE Edit Menu, there is one dynamic menu item that is applicable to the Report Designer.

If the **Report Designer** is opened:

Exit Structure Designer (CTRL+E)
Exits the **Report Designer**. You are prompted to save or discard any changes.

If the Report Designer is closed:

Structure Designer (CTRL+D)
In the editor, place your cursor inside any REPORT structure and select this item. The **Report Designer** will be opened.

Report Designer Menu

Band View
Lets you edit your report and place controls separately, in each individual band.

Page Layout View
Lets you reposition and resize your report bands by dragging handles. All bands display together on a representation of the page.

Properties
Opens the **Properties** pad for the selected control or report band.

Populate Menu

Columns
Lets you place a string variable control tied to a data dictionary field or memory variable. The **Select Field** dialog appears. Select a field or variable, then CLICK in the window.

Multiple Columns
Lets you place a string variable control tied to a data dictionary field or memory variable. The **Select Field** dialog appears. Select a field or variable, then CLICK in the window.

After placing the first field, the **Select Field** dialog appears again, ready for you to place another field. When all fields are placed, press the **Cancel** button in this dialog to return to normal editing.

Report Designer > Format

The Format menu provides commands for spacing and sizing the controls within the report. You may place two or more controls so that their 'edges' match up with each other. You may also spread the controls out, or make all of them the same size.

To do so, first select two or more controls. Select the first by clicking on it. Select the second and subsequent controls by pressing the CTRL key, then clicking on the second control while the shift key remains pressed.

Align > Lefts	Aligns the left borders of the selected controls with the left border of the last control selected (black handles).
Align > Centers	Aligns the center of the selected controls with the horizontal center axis of the last control selected (black handles).
Align > Rights	Aligns the right borders of the selected controls with the right border of the last control selected (black handles).

Align > Tops	Aligns the top borders of the selected controls with the top border of the last control selected (black handles).
Align > MIddles	Aligns the center of the selected controls with the vertical center axis of the last control selected (black handles).
Align > Bottoms	Aligns the bottom borders of the selected controls with the bottom border of the last control selected (black handles).
Align > To Grid	Snaps the selected controls to the nearest grid coordinate.
Make Same Size > Width	Makes all selected controls the same width as the last control selected (black handles).
Make Same Size > Size To Grid	Makes all selected controls the same size as the nearest grid points.
Make Same Size > Height	Makes all selected controls the same height as the last control selected (black handles).
Make Same Size > Both	Makes all selected controls the same width and height as the last control selected (black handles).

Format >Horizontal Spacing

Make Equal	Equalizes the horizontal spaces between the selected controls.
Increase	Increases the horizontal spaces between the selected controls.
Decrease	Decreases the horizontal spaces between the selected controls.
Remove	Removes all horizontal spaces between the selected controls.

Format > Vertical Spacing

Make Equal	Equalizes the vertical spaces between the selected controls.
Increase	Increases the vertical spaces between the selected controls.
Decrease	Decreases the vertical spaces between the selected controls.
Remove	Removes the vertical spaces between the selected controls.

Format > Center in Form

Horizontally	As a group (relative positions of selected controls don't change), centers the selected controls vertically within the window.
Vertically	As a group (relative positions of selected controls don't change), centers the selected controls horizontally within the window.

Format > Order

Bring to Front	For overlapping controls, select this item to move the active control to the top.
Send to Back	For overlapping controls, select this item to move the active control to the back.

Format > Lock Controls

Lock Controls	This option locks all controls to their current position. A "locked" icon appears in the upper left area of the control. When a control is locked, it may not be moved or resized.

Report Designer > Bands Menu

Page Header
Adds a header band to your report.

The HEADER structure traditionally prints at the top of each page of the report. Typically, you place the report title, graphics and other "introductory" elements in the HEADER.

Page Footer
Adds a footer band to your report.

The FOOTER structure traditionally prints at the bottom of the report. Typically, you place a page number, or totals in the FOOTER.

Report Form
Adds a form band to your report.

The FORM structure prints as a "background layer." Typically, you may display "overlays" such as graphics and field labels in the FORM layer, then print the actual foreground data in the DETAIL. The FORM remains constant from page to page.

Detail
Adds a detail band to your report..

The DETAIL structure is the "body" of the report. It contains the basic data, either in table or record format.

Break Group
Adds a new detail, break, group header and group footer bands. Place the crosshair where you want the new group of bands to appear, and CLICK. The **Break Properties** dialog appears. Specify the variable to break on and press **OK.**

A Group BREAK structure can have nested BREAK structures, each with their own HEADER, DETAIL, and FOOTER structures.

See Also:
How to Set Report Group Breaks
How to Sort Reports

Group Header
Adds a new Group Header band to the selected break section.

Group Footer
Adds a new Group Footer band to the selected break section.

Surrounding Break
Adds a break group around an existing detail. Place the crosshair on the detail you want to break on, and CLICK. The **Break Properties** dialog appears. Specify the variable to break on and press **OK.**

View Menu

Expand Bands
Expands or contracts all report bands at once.

Preview!

Opens the **Preview Print Details** dialog which lets you generate "filler" data for your report. The data have no values, but serve as placeholders, so you can get a feel for the appearance of your finished report. Fonts, sizes, colors, and positions of report controls are all displayed

You can quickly "preview" alternative layouts for DETAILs, HEADERs, and FOOTERs, and you can see the effects of the page breaking options you have chosen, all without actually compiling or running your report. **See Also:** Using Print Preview.

Clarion Report Controls ToolBox

When the Report Designer is opened, additional controls are placed in the Report Designer via the Report Controls ToolBox.

To open the Report Controls ToolBox, select **View ▸Toolbox**, or press **CTRL + ALT + X**.

To populate a control simply click on the control name in the toolbox, and drag it to the target location.

Box	Lets you place a BOX control in the selected band. **See Also:** Box Control Properties .
Check Box	Lets you place a CHECK control in the selected band. **See Also:** Check Box Control Properties .
Ellipse	Lets you place an ELLIPSE control in the selected band. **See Also:** Ellipse Control Properties .
Group Box	Lets you place a GROUP control (group box) in the selected band. **See Also:** Group Control Properties .
Image	Lets you place an IMAGE control (graphic image) in the selected band. **See Also:** Image Control Properties .
Line	Lets you place a LINE control in the selected band. **See Also:** Line Control Properties
List Box	Lets you place a LIST control (list box) in the selected band. **See Also:** List Control Properties .
Option Box	Lets you place an OPTION control (OPTION structure, which appears as a group box with radio buttons) in the selected band. **See Also:** Option Control Properties .
Radio Button	Lets you place a RADIO control in the selected band. **See Also:** Radio Button Control Properties .
String	Lets you place a STRING control on the selected band. **See Also:** String Control Properties .
Text Field	Lets you place a TEXT control on the selected band. **See Also:** Text Box Control Properties .

Report Properties

This dialog lets you set up the basic report options, including its page orientation, measurement units, margins, and paper size.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Report Designer** adds the COLOR attribute to your report declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" all the controls on the report so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Extra

Preview	Specifies the name of a QUEUE which stores the filename(s) (*.WMF) for the metafile(s) generated for page preview. See the PREVIEW attribute. If you are using the Report Template, it is handled automatically if you check the Print Preview box and you should leave this entry blank.

General

Jobname	Names the print job, as listed in the Windows Print Manager application.
Label	Type a valid Clarion label to name the REPORT data structure.
Landscape	Specifies landscape paper orientation. New reports default to portrait mode. Landscape means the report text is aligned parallel with the longest paper edges.
Layout	Indicates the orientation of the report controls. **Left to Right** maintains the original layout specified in the Report Designer. **Default** field navigation moves from left to right. **Right to Left** essentially "flips" the report controls' display as a mirror image of the layout specified in the Report Designer.
Paper-Type	Choose from over 40 standard sizes, or choose ***Other*** to specify a custom size.
Width	Specifies a custom paper width in units specified on the General tab.
Height	Specifies a custom paper height in units specified on the General tab.
Prefix	Specifies the label prefix for the REPORT structure.

TextFont To set the default font for all controls appearing in the report, press the **Font** button, then choose the font and style in the Font dialog. You may override the default by setting a different font in the Properties dialog for any specific control. The options you choose in the dialog become the parameters for the FONT attribute. As you choose options, the dialog box displays a sample of the formatting.

Units Specifies the default measurement for all controls placed in the report. Choose **Dialog** Units, **THOUS**andths of Inches, **M**illi**M**eters or **POINTS**.

Position

Lets you set the location and size of the report detail print area, by filling in the AT attribute. The measurement units for these boxes are specified on the **General** tab.

To set a precise starting point for your print detail area relative to the top left corner of the paper, specify **Top Left Corner** coordinates with this dialog. In effect, this establishes the left **margin** for your report. The top margin is usually determined by the Page Header position. These settings may also be accomplished visually by dragging report sections and borders in **the Report Designer's Page Layout View**.

To set the size of the print detail area, choose from the following options for the **Width** and **Height**. When changing a report from portrait to landscape, or vice versa, you should also change the width and height on this tab.

Default Sets a value based on the Paper **Type** property.

Value To set a specific size, mark the **Value** choices.

See Also: Printing Labels

See Also: How to Use the Report Designer -- An Overview

Quick Links

Embeds Accesses the **Embedded Source** dialog for this procedure.

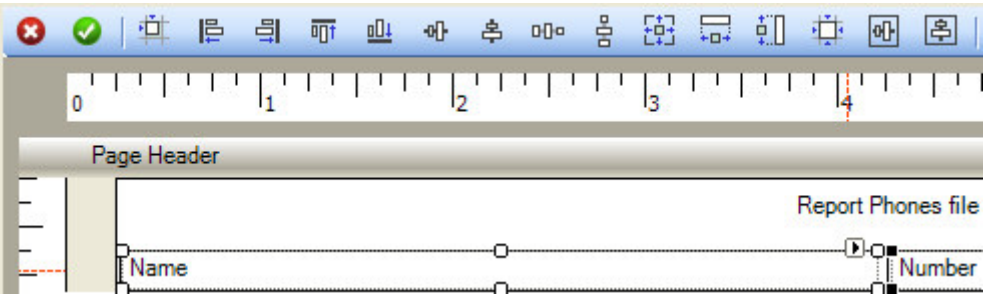
Using the Report Designer

Report Designer Alignment Options

The **Report Designer's Alignment** options allow you to quickly, professionally, and precisely align the controls in your report.

Alignment options discussed below are located from the main IDE menu in the **Report Designer** ▸ **Format** menu.

You can also access the Report Designer's alignment options using the Alignment Bar located at the top of the Designer workspace:



Alignment Bar

Select the controls to align (CTRL+CLICK allows you to select multiple controls, or you can "lasso" multiple controls with CTRL+DRAG), then click on the appropriate alignment tool. All the alignment actions are also available from the **Alignment** menu.

Tip

For most alignment functions, the first controls selected (white handles) are aligned with the last control selected (black handles). That is, the last control selected is the anchor control. It doesn't move, the others do.

Tip

Position the cursor over any alignment button and wait for half a second. A tool tip appears telling you the type of alignment this tool will accomplish.

Align to Grid	Aligns the upper left corner of the controls to the nearest grid point.
Align Lefts	Aligns the left borders of the selected controls with the left border of the last control selected (black handles).
Align Rights	Aligns the right borders of the selected controls with the right border of the last control selected (black handles).
Align Tops	Aligns the top borders of the selected controls with the top border of the last control selected (black handles).
Align Bottoms	Aligns the bottom borders of the selected controls with the bottom border of the last control selected (black handles).
Align Middles	Aligns the horizontal middles of the selected controls with the horizontal middle of the last control selected (black handles).

AlignCenters 	Aligns the vertical middles of the selected controls with the vertical middle of the last control selected (black handles).
Spread Horizontally 	Equalizes the horizontal spaces between the selected controls.
Spread Vertical 	Equalizes the vertical spaces between the selected controls.
Make Same Size 	Makes all selected controls the same height and width as the last control selected (black handles).
Make Same Width 	Makes all selected controls the same height as the last control selected (black handles).
Make Same Height 	Makes all selected controls the same height as the last control selected (black handles).
Size to Grid	Expands all corners of the selected controls to the nearest grid point.
Center Horizontally 	As a group (relative positions of selected controls don't change), centers the selected controls vertically within the band.
Center Vertically 	As a group (relative positions of selected controls don't change), centers the selected controls horizontally within the band.

Using the Report Designer - Command Toolbar

The **Report Designer** contains a **Command** toolbar. The toolbar lets you quickly execute a variety of **Report Designer** functions at the touch of a button.

All the commands in the **Command** toolbar are also available from the IDE menu.

Position the cursor over any tool and wait for half a second. A tool tip shows you the type of control this tool creates.

Cancel
Exit the Report Designer and abandon changes.

Save and Close
Exit the Report Designer and save changes.

Bring to Front
For overlapping controls, select this item to move the active control to the top.

Send to Back
For overlapping controls, select this item to move the active control to the back.

Tab Order
Not applicable in the Report Designer.

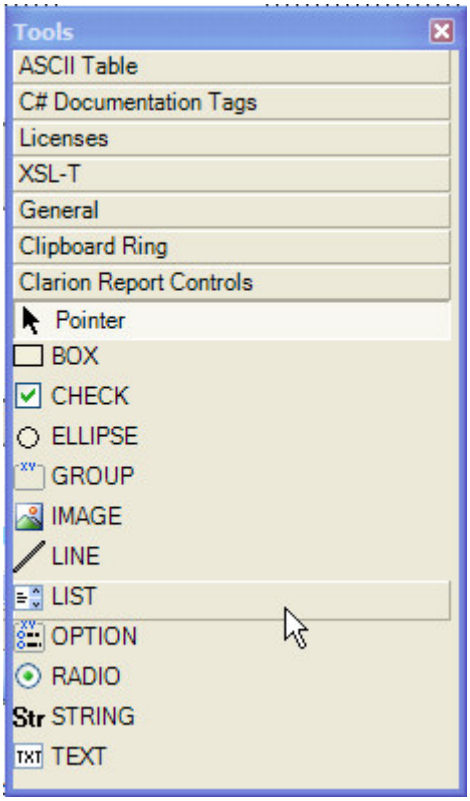
Print Preview
Calls the Print Preview dialog, allowing to you render a sample report for viewing.

Switch SupressTransparency
Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Using the Report Designer - Controls Toolbox Pad

The **Report Designer** contains a floating **Controls** toolbox, similar to the **Window Designer**. Simply choose a control from the toolbox (CLICK on it), then CLICK in the report band (or DRAG to the target location) to place the control in the report.

Display or hide the **Controls** toolbox by choosing **View** ▸**Tools.** Resize the **Controls** toolbox by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG. All the controls in the toolbox are also available from the **Controls** menu.



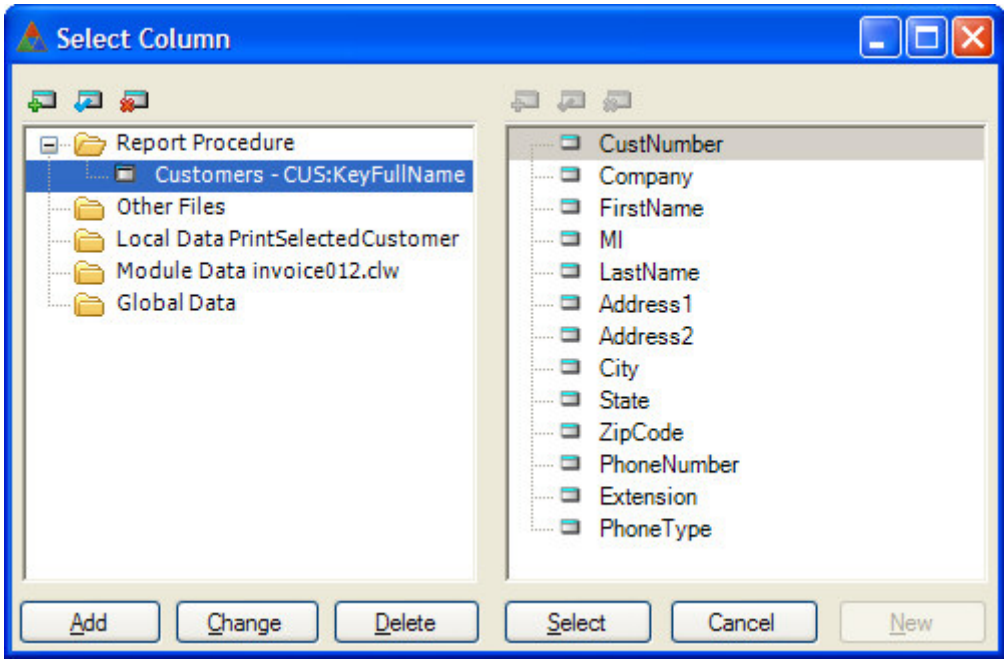
BOX	Allows you to place a BOX control on the report under construction.
CHECK	Allows you to place CHECKBOX control on the report under construction.
ELLIPSE	Allows you to place ELLIPSE control on the report under construction.
GROUP	Allows you to place GROUP control (group box) on the report under construction.
IMAGE	Allows you to place IMAGE control (graphic image) on the report under construction.
LINE	Allows you to place LINE control on the report under construction.
LIST	Allows you to place LIST control (list box, or drop down list box) on the report under construction.
OPTION	Allows you to place OPTION control (OPTION structure, which appears as a group box with radio buttons) on the report under construction.

RADIO	Allows you to place RADIO control on the report under construction.
STRING	Allows you to place STRING control on the report under construction.
TEXT	Allows you to place TEXT control on the report under construction.

Using the Report Designer - Populate Columns

The **Report Designer** contains a **Populate Columns** option. This allows you to quickly "populate" a window with entry controls and prompts for columns in your data dictionary table.

Open the **Populate Columns** dialog by choosing **Report Designer ▸Populate ▸Column.** Resize the **Populate Columns** dialog by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG.



1. Choose a **table** from the list in the left pane.
2. CLICK on the **column** you want on your report.
3. Press the **Select** button.
4. CLICK in the report band to place the control.

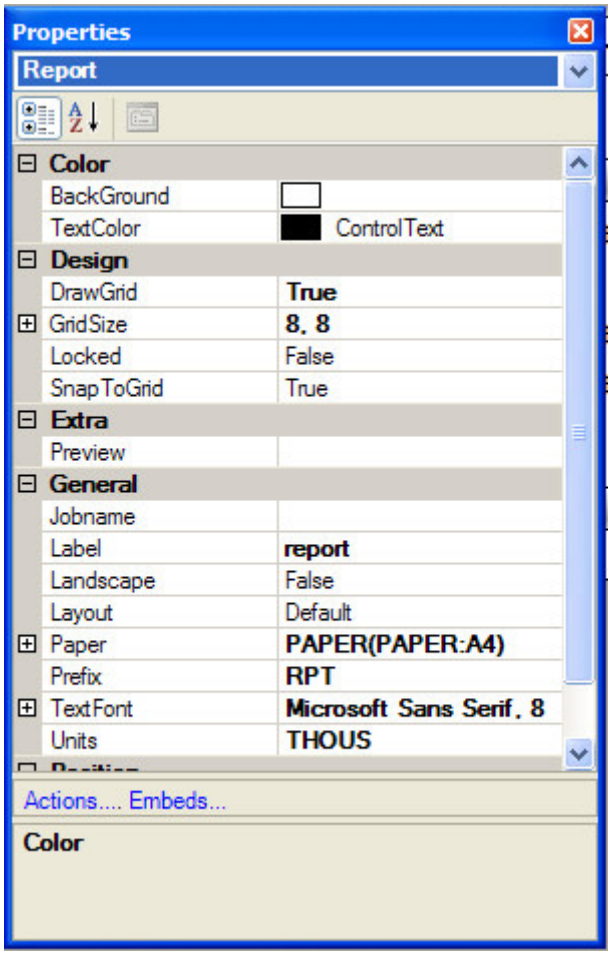
The cursor becomes a crosshair. The top left corner of the control is placed at the intersection of the cursor crosshair.

The **type** of control (text box, check box, radio button, etc.) is determined by the settings for this particular column in the Data Dictionary **Column Properties** dialog.

Using the Report Designer - Properties Pad

The **Report Designer's Properties Pad** allows you to quickly specify the appearance and content of all properties in the report.

Display or hide the **Properties Pad** by choosing **View ▸Properties** (or simply press the F4 key). Resize the **Property** toolbox by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG.



Using the Report Designer - Sample Reports

The **Report Designer** is a visual design tool. In **Band View**, you always see a sample of the report band you're working on, as you work on it. For example, place a list box in the detail band and drag its handles to the size you want.

Switch to **Page Layout View** to resize and reposition the report bands. Drag the band handles or drag the entire band. All bands appear together on a representation of the page.

In addition, you can quickly generate filler data and see a sample report by choosing **Preview!** from the action bar, all without actually compiling or running the report.

ReportWriter

ReportWriter for Clarion 7

ReportWriter for Clarion 7 is installed in the \Clarion7%ROOT%\ReportWriter directory.

It can be accessed within the IDE from the **Tools** menu, or externally from a shortcut in your Start Menu Program Group.

Importing Dictionary Information

ReportWriter works with the Clarion Dictionary's **TXD** format.

You can export your Clarion7 DCT to TXD format by choosing **Export to text** in the DCT Explorer toolbar. When saving, change the extension to *.TXD* in the *Save File* dialog

The Clarion 7 dictionary support for TXD export is provided solely for compatibility with Report Writer and is "export only". Except for use with ReportWriter you should use the new DCTX XML format when working in Clarion 7. You cannot import a .TXD file created in Clarion 7, and an error is reported if you attempt to do so.

More information regarding ReportWriter in general can be found in the ReportWriter core help file included in this Clarion 7 install.

Special IDE Features

IDE Code Regions

!region <labelName>
<some source text>
!endregion

!region	Begins a foldable code region
labelName	An optional text description that displays when the code region is folded
!endregion	Terminates a foldable code region

A code region lets you specify a block of code that you can expand or collapse when viewing it in the Clarion# IDE.

Example:

Here is how a code region looks in the IDE:

```
8         MEMBER('')
9       !region NAMESPACE and USING
10        NAMESPACE('Threads')
11        USING('System')
12        USING('System.Drawing')
13        USING('System.Windows.Forms')
14      !endregion
15        !!! <summary>
16        !!! Description of ${ClassName}.
17        !!! </summary>
```

Region Expanded

```
8         MEMBER('')
9       NAMESPACE and USING
15        !!! <summary>
16        !!! Description of ${ClassName}.
17        !!! </summary>
```

Region Folded

XML Documentation Comments

This type of comment provides the ability to generate XML documentation for Clarion .Net source files. (For more information see XML Documentation in MSDN). All comments started with "!!!" are considered as documentation comments.

Example:

```
!!!<summary>
!!! This is a class A
!!!</summary>
A  CLASS
    …
   END
```

Quick XML Documentation

The Quick XML Documentation feature is only available in Clarion.NET.

To write a documentation comment to any target object, start the comment with 3 exclamation marks:

For example:

```
!!! <summary>
!!! Special Formatting Documentation
!!! </summary>
!!! <param name="b">The parameter comment</param>
!!! <remarks>
!!! This is a remark.
!!! </remarks>
class.proc  procedure(long b)
   code
```

If you use XML-styled documentation comment syntax (as in the above example) you will see special formatting in the tooltip text provided by the Code Completion. Otherwise, the comments will be displayed in the tooltip "as is".

Documentation comments must precede the object declaration. For procedures, you can have comments for the procedure declaration or procedure definition (the comment for declaration has higher priority).

Other examples:

```
test  long !comment to test
test1 long
```

In the example above, *test1* will have a "comment to test" documentation comment. The *test* variable will not, since it comes before the *test1* variable declaration.

```
! comment1
c class
p procedure !proc comment
end
```

In this example, the *p* procedure will have no documentation comment, but the *c* class will display both comment lines *comment1* and *proc comment*.

Structure (Window) Designer

Alert Keys Dialog

This dialog lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

To specify the first Alert key:

Add Press the **Add** button. Specify the key or key combination with the **Input Key** dialog. The key combination appears in the **List**. Repeat for any additional Alert keys.

Remove To delete a key combination, highlight the key combination in the **List**, then press the **Remove** button.

It's up to you to add code to "do something" upon detecting the EVENT:AlertKey.

Window Designer - Alignment Toolbar

The **Window Designer's Alignment** toolbar allows you to quickly, professionally, and precisely align the controls in your window.

Select the controls to align (CTRL+CLICK allows you to select multiple controls, or you can "lasso" multiple controls with CLICK+DRAG), then click on the appropriate alignment tool. All the alignment actions are also available from the **Window Designer** > **Format** menu.

Tip

For most alignment functions, the last set of controls selected (black handles) are aligned with the first control selected (white handles). That is, the first control selected is the anchor control. It doesn't move, the others do.

Tip

Position the cursor over any tool and wait for half a second. A tool tip appears telling you the type of alignment this tool will accomplish. From left to right on the toolbar:

Align to Grid	Aligns the upper left corner of the controls to the nearest grid point.
Align Lefts	Aligns the left borders of the selected controls with the left border of the target alignment control selected (white handles).
Align Rights	Aligns the right borders of the selected controls with the right border of the target alignment control selected (white handles).
Align Tops	Aligns the top borders of the selected controls with the top border of the target alignment control selected (white handles).
Align Bottoms	Aligns the bottom borders of the selected controls with the bottom border of the target alignment control selected (white handles).
Align Middles	Along a horizontal axis, aligns the centers of the selected controls with the center of the target alignment control selected (white handles).
Align Centers	Along a vertical axis, aligns the centers of the selected controls with the center of the target alignment control selected (white handles).
Spread Horizontally	Equalizes the horizontal spaces between the selected controls.
Spread Vertically	Equalizes the vertical spaces between the selected controls.
Make Same Size	Makes all selected controls the same height and width as the target alignment control selected (white handles).
Make Same Width	Makes all selected controls the same width as the target alignment control selected (white handles).
Make Same	Makes all selected controls the same height as the target alignment

Height	control selected (white handles).
Size to Grid	Expands all corners of the selected controls to the nearest grid point.
Center Horizontally	As a group (relative positions of selected controls don't change), centers the selected controls vertically within the window.
Center Vertically	As a group (relative positions of selected controls don't change), centers the selected controls horizontally within the window.

Additionally, there are other helper buttons located on the Alignment toolbar:

Bring to Front	For overlapping controls, select this item to move the active control to the top.
Send to Back	For overlapping controls, select this item to move the active control to the back.
Tab Order	This option lets you visually specify the tab-stop order of the controls in the window. A small box with a number inside appears on each control, indicating the current order. CLICK on the controls to change the order to the order you wish. To cancel the tab order change, simply right-click outside of the window area and toggle this item off. Parent/Child controls are marked with the following convention: *parent control.child control.grand-child control* For example, a STRING that is populated on a SHEET/TAB control may show *1.0.2*. The second SHEET on the window, first TAB in that sheet, and the third control populated on that tab This feature uses the Microsoft Forms standard, where controls are number starting at control zero (0) as the first control.
Print Preview	Calls the Print Preview dialog. From here, you can review a file's contents before printing (Report Designer only).
Switch SuppressTransparency	This button enables the design time **SuppressTransparency** property. The effect of this property is to allow the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. It has no effect on the actual runtime window display.
Switch UseVisualStyles	This button controls the display of Visual Styles for all valid controls in design mode (controls appear as if the XP manifest was active).

Application Properties Dialog

This dialog lets you specify the appearance and functionality of your application frame window. (The APPLICATION structure)

Color

Enter a valid color equate in the **TextColor** field, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your window declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

DrawGrid Set to True to display the design grid, or set to False to turn it off.

GridSize Grid values are specified in dialog units, a unit of measure based on the current system font. Enter the horizontal and vertical grid size, separated by commas. **Width** is the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on. **Height** is the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

Locked "Freezes" all the controls on the window so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

SnapToGrid Set to True to turn grid snap on; set to False to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

Extra

Icon To associate an icon with the window, specify an icon in this field. You may type in a file name or an EQUATE. You may also press the ellipsis button (...), then select an icon file name using the standard Open File dialog. The file name or equate you specify becomes the parameter for the ICON attribute.

You should always specify an icon for an application window. Specifying an icon name automatically places a minimize button on the caption bar of your application or MDI child window.

Palette Use the PALETTE attribute to specify maximum color depth. The PALETTE attribute specifies how many colors you want this window to use when it is the foreground window. For example, 24-bit color would be 16777215. The number you specify becomes the parameter for the PALETTE attribute. Leave this field blank to specify the default for the end user's system.

Status Bar

To provide a message bar at the bottom of your window, mark the **Status Bar** check box. This adds the STATUS attribute to the window.

A status bar in an application window is an excellent way to increase user feedback in your application. Clarion makes it simple to post messages on the status bar advising the user of what your application is doing as it does it. Increasing user feedback makes the user feel more in control. This allows the user to feel more confident and be more efficient when using your application.

Status Widths

To set the width of the status bar zone(s), type a value or list of values in the **Status Widths array** field.

You must set the **Statusbar Status** field to True to display a status bar.

Press the ellipsis to access the INT32 Collection Editor. This dialog allows you to easily add additional status bar sections and their corresponding widths.

The values you enter in this field fill the STATUS attribute parameters.

The zones are the areas within the status bar marked off by the 3D shaded boxes. The first zone on the left, by default, displays MSG attribute text. This is useful for specifying short help instructions or other information to the user. If your application has only one zone for the status bar, you may omit this field. For more than one zone, enter additional array values.

The default measurement unit is dialog units.You may set a minimum value for a zone width by typing a negative number. This creates a zone with a minimum width, but is expandable by resizing the window. Use the runtime property assignment syntax to place text in any zone. To place a string in the second zone, for example:

{PROP:StatusText} = ?array

A multi-zone status bar can give your application a professional look. You may display help text in zone one, and when editing a record, the current record number in zone two, for example.

Timer

To specify the window receive Timer Event messages from Windows, fill in the **Timer** field. Specify the timer interval in hundredths of seconds. The file name or equate you specify becomes the parameter for the TIMER attribute.

For example, if you specify 100 in the field, the window will automatically receive an EVENT:Timer once every second (100/100's seconds). This might be appropriate for adding a clock to a status bar.

General

Frame Type To choose the frame for your window, pick a selection from the **Frame Type** drop-down list. The frame defines the borders of the window. The normal frame type for an application frame is the resizeable type. Choose from:

Single – a single pixel frame which the user cannot resize.

Double – a thick frame, which the user cannot resize. This adds the DOUBLE attribute to the window.

Resizeable - a thick frame, which the user can resize. This adds the RESIZE attribute to the window.

Initial Size Sets the initial state of your window. Choose from:

Normal - displays the window at the default size which either you specifically set, or Windows sets if you don't.

Maximized - the window fills the desktop, if an application window, or the window frame, if an MDI child window. This adds the MAXIMIZE attribute to the window.

Iconized - the window appears in iconized state--as a 32 by 32 pixel window at the bottom of the desktop. This adds the ICONIZE attribute to the window.

Label To specify the label for the application structure, type it in the **Label** field. This names the specific APPLICATION in the source code. The label may contain upper or lower case letters, numerals, the underscore character or a colon. Space characters are forbidden. The first character must be a letter or the underscore character. Clarion reserved words may not serve as labels.

Layout Indicates the orientation of window controls and field sequence.

Left to Right maintains the original layout specified in the Window Designer.

Default field navigation moves from left to right.

Right to Left essentially "flips" the window controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left.

The setting in the Application Frame will cascade its setting to all child window that have the **default** setting active.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the

dialog box displays a sample of the selected font.

Title To specify caption bar text, type a string constant in the **Text** field. The caption bar holds the name of the window.

Wallpaper-BackgroundImage To provide a background image for the window's client area, specify an image filename. Type the filename or press the ellipsis button (...) to select a file. See WALLPAPER in the *Language Reference*.

Mode Specify how the window displays the background image. Choose from:

Stretched The image expands to fill the entire client area.

Centered The image displays at its default size and is centered in the window's client area.

Tiled The image displays at its default size and is repeated so it fills the entire client area.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Options

AutoDisplay To add the AUTO attribute to your window, set **AutoDisplay** to True. This automatically updates the contents of all controls on screen through each pass of the ACCEPT loop.

EntryPattern To enable support for an entry mask for controls in the window, set **EntryPattern** to True. This lets you specify key-in entry patterns for the fields you choose, and adds the MASK attribute to the window.

Immediate To generate a message event each time the end user moves or resizes the window, set **Immediate** to True. This adds the IMM attribute to the window. You are responsible for the code that executes upon notification of the event.

MaximizeBox To place a maximize button in your window, set **MaximizeBox** to True. In general, you should place a maximize button only on application windows and MDI child document windows. This adds the MAX attribute to the window.

SystemMenu To add a system menu to your window, set **SystemMenu** to True. Most windows should have a system menu. For users on a system without a mouse, the system menu provides the only means of minimizing, maximizing or re-sizing the window. This adds the SYSTEM attribute to the window.

Even if you plan that the window should NOT have a system menu when the application is complete, it's good practice to place a system menu on your application while it's under development. By DOUBLE-CLICKING the system menu, or choosing Close, you can close your application should your normal exit procedure fail.

Position

Lets you set the location and size of a window.

Position lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the size visually by dragging with the mouse in the **Window Designer**.

To set the location of the window's **Top Left Corner**, choose from the following options for the **X** (horizontal) and **Y** (vertical) coordinates.

Center Centers an APPLICATION window on the screen. Centers child windows on their respective parents.

Default The Windows operating system determines the initial position of the window. By default, Windows position windows in a cascading sequence from top-left to bottom-

right.

Value Lets you set a precise coordinate in Dialog units. Generally, the coordinate is relative to the top left corner of the screen for APPLICATION (first or main) windows and relative to the top left corner of the APPLICATION window for all subsequent (child) windows.

To set the window's size, choose from the following options for the **Width** and **Height**.

Default The Windows operating system determines the initial size of the window.

Value Lets you set a precise width or height in Dialog units. Dialog units provide a relative screen measure based on the system font character size.

Scrollbars

To add a horizontal scroll bar to your window, set **Horizontal** to True. Scroll bars only appear when something inside the window--a control--is bigger than the window. To add a vertical scroll bar to your window, set **Vertical** to True. These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the window.

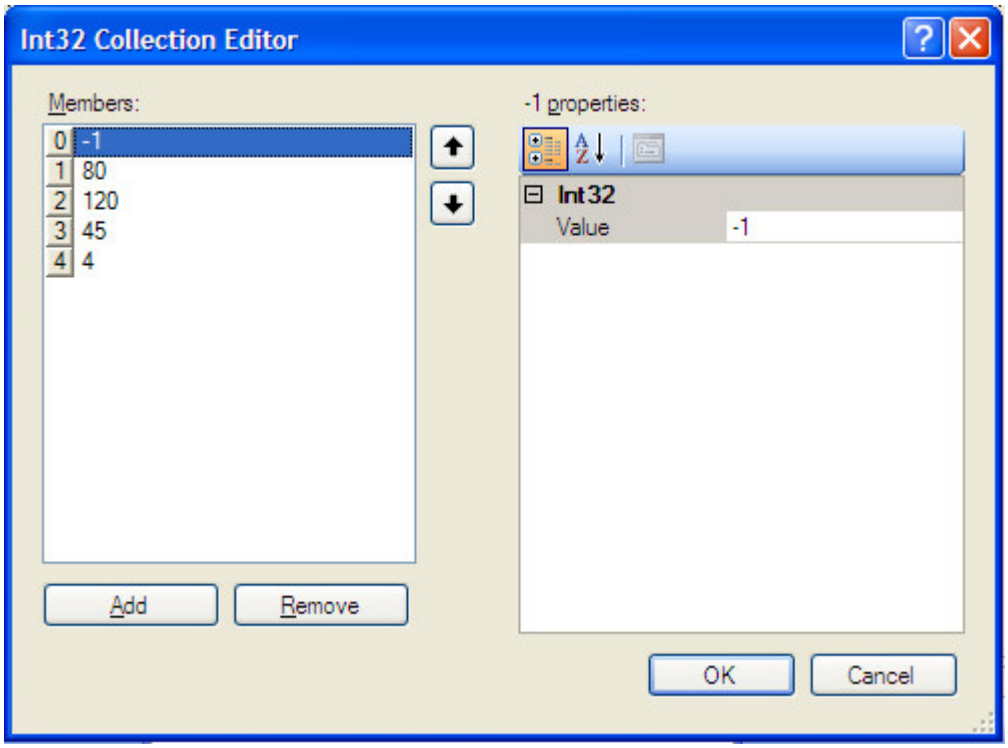
Assuming your application frame will display MDI child windows, you normally set both **Horizontal** and **Vertical**.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this procedure.

Collection Editor

The INT32 Collection Editor is a dialog window that is designed to allow you to add and modify selected arrayed properties.



The *Members* list displays the active array elements. Press the Add button to add a new array element. Highlight an existing array elelemnt and press the Remove button to delete the element.

The *Properties* list lets you add a specific value, and other property attribures when appropriate.

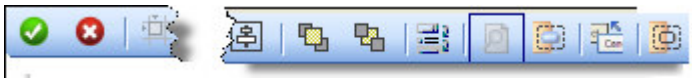
Window Designer - Command Toolbox

The **Window Designer** shares a **Command** toolbar with the **Alignment** toolbar. The toolbar lets you quickly execute a variety of **Report Designer** functions at the touch of a button.

All the commands in the **Command** toolbar section are also available from the IDE menu.

Tip

Position the cursor over any tool and wait for half a second. A tool tip shows you the type of control this tool creates.



Save and Close
Exit the Report Designer and save changes.

Cancel
Exit the Report Designer and abandon changes.

Bring to Front
For overlapping controls, select this item to move the active control to the top.

Send to Back
For overlapping controls, select this item to move the active control to the back.

Tab Order
Not applicable in the Report Designer.

Print Preview
Calls the Print Preview dialog in the Report Designer, allowing to you render a sample report for viewing.

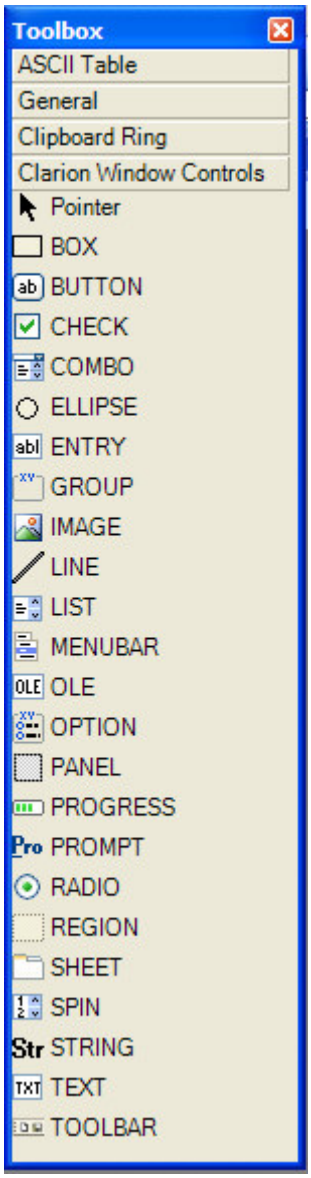
Switch SupressTransparency
Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Use Visual Styles
This button controls the display of Visual Styles for all valid controls in design mode (controls appear as if the XP manifest was active).

Hide Hidden Controls
Press this button to hide all controls in the Designer that has the **Hide** property set to TRUE. This allows for a better visual display, particularly when there are overlapping controls.

Window Designer - Controls Toolbox

The **Window Designer** contains a floating Controls toolbox, similar to those found in many draw or paintbrush programs. Simply choose a control from the toolbox (CLICK on it), then DRAG to the sample window at the target location to place the control in the window.

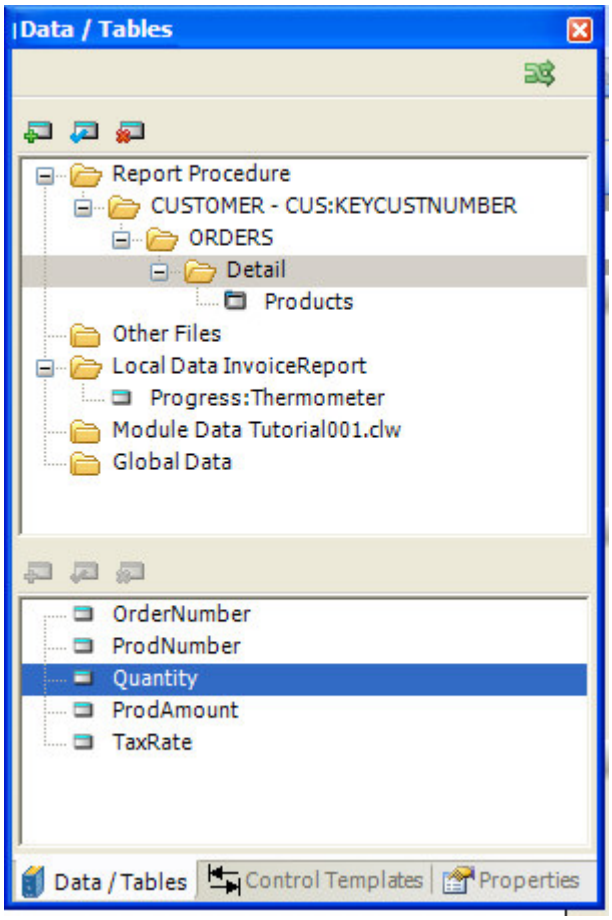


Display or hide the **Controls** toolbox by choosing **View** ▸**Toolbox.** Resize the **Controls** toolbox by placing the cursor on the border of the box. Select a control by clicking on it, and DRAG to the desired location on the target window.

Window Designer - Data/Tables Pad

The **Window Designer** has access to the **Data / Tables Pad**. This pad allows you to quickly "populate" a window with entry controls **and** prompts for fields in your data dictionary tables.

Display or hide the **Data / Tables Pad** by choosing **View ▸Data / Tables Pad.** Resize the **pad** by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG.

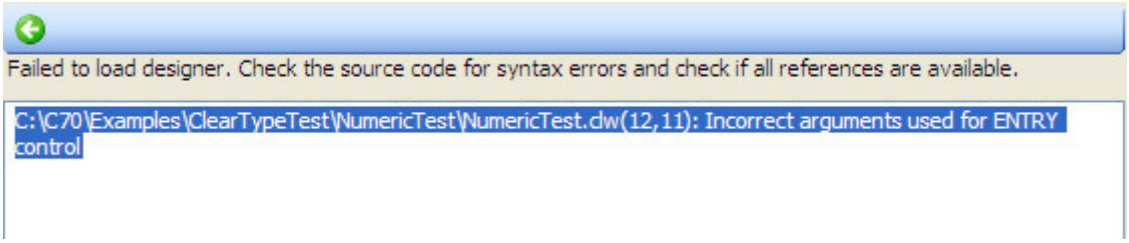


1. Select a **table** from the top pane.
2. Select the **column** you want on your window in the bottom pane.
3. DRAG the column in the sample window to place the control **and** its associated prompt.

The **type** of control (entry box, check box, radio button, etc.) is determined by the settings for this particular column in the Data Dictionary.

Designer Error Window

The Designer Error window is a special window that provides specific information in the event that a window or report structure could not be loaded for any reason.



In the Clarion 7 Designers, the most likely reason is a missing or incorrect element contained within the referenced structure.

In the Clarion# designers, the most likely cause id a missing component or declaration in the Clarion# project source.

Usually, the information presented will refer you to a precise line in the source from where the problem originated.

Font Dialog

Lets you change the font, style (such as bold and italic), font size, color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the formatting.

Choose from the following:

Font Type or select a font (typeface) name. The dialog lists the fonts available with the current printer driver and additional fonts installed in your system.

Font Style Select a style. To use the default type style for a given font, select Regular. Depending on the fonts installed, you can choose bold, italic, or bold italic.

Size Type or select a size. The sizes available depend on the printer and the selected font.

Effects Select the formatting options you want. Choose:

Underline - underlines all characters, including the spaces between words, with a single line.

Strikeout - draws a line through selected text.

Color Type or select one of the sixteen predefined colors. To display color, you must have a color monitor; to print color, you must have a color printer.

Input Key Dialog

Use this dialog to specify a key, or key combination, for a hot key (KEY attribute) or an alert key (ALRT attribute):

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control.

The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Modifiers Optionally, add additional keys to your key sequence by checking the **Ctrl**, **Alt,** or **Shift** boxes, or any combination of the three.

Mouse Mouse clicks may be used within the key sequence; however, mouse clicks *cannot* be specified by clicking the mouse. For mouse clicks, check the corresponding check box(es). For example, to act on a double-click, check the **Left Button** box *and* the **Double Click** box.

List Box Formatter Dialog

The List Box Formatter dialog shows how the list box under construction looks. It fills this sample list box with placeholder characters representing the contents of each field. If any field contains a header, a header row appears over the column.

You format the fields one by one in the **Properties** dialog. The sample list box always displays a horizontal scroll bar, whether you specify one in the **List Properties** dialog or not.

Select multilple columns for editing using CTRL + CLICK. You can also just use the keyboard. Hold the CTRL key and press the up or down arrow key. To unselect all items just release the Ctrl key and press the up or down arrow key. or release the CTRL key and mouse click on any column.

The formatter does *not* display a vertical scroll bar. If the queue contains more items than rows in the list, and if you add the VSCROLL attribute by checking the box in the **List Properties** dialog, the vertical scroll bar appears at run time.

The dialog contains the following buttons:

Add Group


To add a new Group, press the **New Group** button (or SHIFT+INS). This adds the currently highlighted column into the new group. Use the **Move Column** butttons to move other columns into (or out of) a group.

Add Field


To add a field to the list box, press the **New Column** button (or INS).

When opening the **List Box Formatter** from within the Application Generator, using a procedure template which supports it, the **Populate** button displays the **Select Field** dialog. From here, you can indicate any database field or memory variable for use as a list box column. The generated code puts the contents of the database records into the queue for use in the list box.

Remove


Removes the currently selected field from the list box.

Move Left


(CTRL + LEFT ARROW) Moves the currently selected field one position to its left. If the selected field is the leftmost in a group, the field moves out of the group, without changing position. If the selected field is immediately to the right of a group, the field moves into the group, without changing position.

Move Right


(CTRL + RIGHT ARROW) Moves the currently selected field one position to its right. If the selected field is the rightmost in a group, the field moves out of the group, without changing position. If the selected field is immediately to the left of a group, the field moves into the group, without changing position.

See the following topics for more information about adding list box functionality:

How to Restore User Resized List Box Column Widths

How to Trap a Double Click on a List Box

How to add Drag and Drop to a List Box

How to Display the Sort Field First on a Multi-Key Browse

Using drop-down lists to Lookup Records

Color and Style

Use these prompts to set the default colors for all list rows and columns.

Background To set the default color for normal (unselected) list background, type a valid color equate in this field, or press the ellipsis (...) button to select a color from the color dialog.

Adds an "E(,*color*,,)" to the FORMAT string.

Default Style Type the default style number. The style number sets the font typeface, size, style, and color for all list rows and columns.

Adds a "Z(*n*)" to the FORMAT string, where *n* is the style number.

Selected Background To set the default color for normal (unselected) list background, type a valid color equate in this field, or press the ellipsis (...) button to select a color from the color dialog.

Adds an "E(,,,*color*)" to the FORMAT string.

Selected Text To set the default color for normal (unselected) list text, type a valid color equate in this field, or press the ellipsis (...) button to select a color from the color dialog.

Adds an "E(,,*color*,)" to the FORMAT string.

TextColor To set the default color for normal (unselected) list text, type a valid color equate in this field, or press the ellipsis (...) button to select a color from the color dialog.

Adds an "E(*color*,,,)" to the FORMAT string.

Data

AutoFieldNumber This property provides support for the Field Number of a column, and the ability to override it. Set this proprty to True to designate that the field numbering of the selected column will be "standard", based on the order that it appears in the list box. Setting this property to False allows the field numbering to be modified. This feature is important for some list box properties that require field number information, and gives you the ability to control it.

DataIndent Optionally specify an indent, in dialog units, for the listbox data. Indent moves the data by the number of dialog units specified, in the opposite direction to the justification. An indent of two (2) on left justified data improves listbox readability. The indent appears within the FORMAT string surrounded by parentheses and preceded by a letter indicating the justification, as in "L(8)."

DataJustification Choose from the drop-down list to specify left, right, center or decimal. Decimal justification aligns decimal numbers by their decimal points. The justification appears in the FORMAT string following the data width, as in "40R."

FieldNumber If the **AutoFieldNumber** property is set to False, enter a custom Field Number here. If the **AutoFieldNumber** property is set to False, this property is Read Only.

Picture Specify the picture token for the data. The **List box Formatter** displays the data according to the picture token. For example, the picture token @P(###) ###-####P displays a phone number as (555) 555-5555.

The picture token you specify appears in the FORMAT string.

Flags

Fixed Set to True to specify that the column always remains visible in the listbox, even if other columns scroll.

The FORMAT string includes the "F" character, immediately preceding the header text as in "F~MyHeader~."

HasColor Set to True to allow conditional runtime colors for individual list items--that is, to conditionally override the default colors for individual list rows. The color information for each row is contained in four LONG fields that immediately follow the data field in the QUEUE. Assign the color value to the appropriate QUEUE field at runtime, and Clarion's runtime library does the rest.

See *Control Templates--BrowseBox Control* for information on specifying conditional BrowseBox colors, and see FORMAT in the *Language Reference*.

Adds an asterisk "*" to the FORMAT string.

LastOnLine Set to True to specify that the next field in the group will appear immediately below the current field (if there are multiple fields in the first line of the group, the field will actually appear below the first field in the first line group). In effect, it stacks two or more fields below the group header.

The field must be part of a group.

For the best display, make sure that your window is not resizable when using multi-line groups.

The FORMAT string includes the "/" character, immediately preceding the header text as in "/~MyHeader~."

Locator By default, the first field in a multi-column COMBO displays in the entry portion of the COMBO. Set **Locator** to True to specify that this field (instead of the first field) displays in the entry box portion of a multi-column COMBO control.

The FORMAT string includes the "?" character, immediately preceding the header text as in "?~MyHeader~."

Resizeable Set to True to specify that the user can resize the width of the columns at run time.

The FORMAT string includes the "M" character, immediately preceding the header text as in "M~MyHeader~."

See How to Restore User Resized List Box Column Widths...

RightBorder Set to True to specify column separators between fields in the listbox at run time.

The FORMAT string includes the pipe symbol (|), immediately preceding the header text, as in "|~MyHeader~."

Style Set to True to allow conditional runtime fonts for individual list items--that is, to conditionally override the default fonts for individual list rows. The font (style) information for each row is contained in a LONG field that immediately follows the data field in the QUEUE. Assign the style value to the appropriate QUEUE field at runtime, and Clarion's runtime library does the rest.

See Control Templates--BrowseBox Control in the Template Guide for information on specifying conditional BrowseBox colors, and see FORMAT in the Language Reference.

Adds a "Y" to the FORMAT string.

Tooltip Set to True to activate a specific tool tip for the selected column. The tool tip content is specified on the Appearance tab. Adds a "P" to the FORMAT string.

Before you can activate individual column tool tips, make sure that you have a tool tip defined for the list box control. This is found on the Help tab of the selected list box control.

Underline Set to True to add the underline style to the listbox text. In effect, this creates a bottom border for each row in the column, giving your listbox a spreadsheet or cell-like appearance.

The FORMAT string includes the underscore character, immediately preceding the header text, as in "_~My Header~."

General

DefaultColumnTip Used to designate default text to be used for the selected column’s tool tip.Adds a "Q" to the FORMAT string.

Icon Select an icon setting from the available drop list:

None Select this to display no icons in the column.

Normal Select this to create an area to the left of the data in the column for displaying a normal image (.ICO) that you supply. See *Control Templates--BrowseBox Control* in the *Template Guide* for information on specifying BrowseBox icons, and see *Prop:IconList* in the *Language Reference*.

Adds an "I" to the FORMAT string.

Transparent Select this to create an area to the left of the data in the column for displaying a transparent image (.ICO) that you supply. See *Control Templates--BrowseBox Control* in the *Template Guide* for information on specifying BrowseBox icons, and see *Prop:IconList* in the *Language Reference*.

Adds a "J" to the FORMAT string.

Header

Indent Optionally specify an indent, in dialog units, for the heading text. Indent moves the data by the number of dialog units specified, in the opposite direction to the justification. An indent of two (2) on left justification improves listbox readability.

This appears within the FORMAT string following the header, as in "~My Header~L(8)."

Justification Choose from the drop-down list to specify left, right, center or decimal header justification.

This appears within the FORMAT string following the header, as in "~My Header~L."

ScrollBar Type a non-zero value to specify a horizontal scroll bar for this column only. If the overall listbox already has a scroll bar, the column scroll bar appears above the listbox scroll bar. The value specifies, in dialog units, how far the column scrolls.

For example, if your *data* is fifty (50) characters, and your listbox *column* width is about forty (40) characters (one hundred sixty (160) *dialog units*), you should specify a value of fifty (50). Fifty (50) additional dialog units are enough to display the ten characters that extend beyond the width of the listbox column.

The scroll bar and size appear in the FORMAT string together, as in "S(4)."

Text Optionally specify header text for the column. The header appears as a gray row above the listbox data items. To specify no header, leave this field blank. If any field included in the listbox has a header, a header appears across the entire listbox; fields with no header text have a blank header.

The heading appears within the FORMAT string enclosed in tilde (~) characters, as in "~My Header~."

Width Specify the width in dialog units for the column data. By default, the Formatter sets the value to four times the number of characters specified in the field picture in the data dictionary. For variables, the default is four times the number

of characters in the picture token defined for it.

As a rough guide, allow four dialog units for an average character. For example, if you want a column 10 characters wide, type 40 in the Width field.

After you've placed a field, you can drag the column separators in the Sample Listbox to resize the column width. The cursor changes when you place it on top of the separator, to indicate you can resize it.

The data width you set appears within the FORMAT string for the field, preceding the Justification code, as in "40L."

Tree

OneBasedTree Set to True to allow the root level to collapse, that is, all the items in the tree can collapse to a single line.

ShowBoxes Set to True to add expand (+) and contract (-) boxes to the tree diagram.

Set to Falser to append "(B)" to the "T" in the FORMAT string, resulting in "T(B)" to suppress boxes.

Adds a "(1)" to the "T" in the FORMAT string, resulting in "T(1)."

ShowLevel Set to True to cause each descending level of the Tree hierarchy to be indented.

Set to False to append "(I)" to the "T" in the FORMAT string, resulting in "T(I)."

ShowLines Set to True to add connecting lines between related items in the tree diagram.

Set to False to append "(L)" to the "T" in the FORMAT string, resulting in "T(L)" to suppress lines.

ShowRoot Set to True to display a root item for the tree diagram.

Set to False to append "(R)" to the "T" in the FORMAT string, resulting in "T(R)" to suppress display of a root item.

Tree Set to True to display this column in a hierarchical tree diagram. See Relation Tree control template.

Adds a "T" to the FORMAT string.

It is not necessary to be concerned with the precise syntax of the List Box Format String. Always use the List Box Formatter to build the string for you. If you are using PROP:FORMAT in your embedded source, you can always cut and paste your format string from the Formatter as needed.

ListBox Styles

A list box style is a collection of fonts and colors that control a list box appearance. Applying a style to a list box is a two-step process. The style attribute must first be enabled for each desired column in the List Box Formatter. Next, a style can be applied in the Actions tab for each column in the list box control. The styles that are applied are maintained and organized in this window.

This dialog provides update buttons to add, change or delete a selected list box style, and is accessed from the **Listbox Styles** button located on the *Procedure Properties* window.

In the subsequent **Styles** dialog, the following prompts are provided:

Each prompt that follows in this dialog has an optional "V" button that, when pressed, provides an ellipsis button for variable selection. This extends the power of listbox styles to allow a given style to be controlled and modified at runtime.

Style Attributes

A list box style contains the following attributes:

Font	Press the **Font** button to designate a **Font Name** (Arial, MS Sans Serif, etc.), **Font Size**, and **Font Style** (Regular, Bold, Italic, etc.) for the active listbox style. Manual entry of these settings is also permitted, but entering incorrect values (Font Names and Styles that are not supported) can produce unpredictable results.
Color	Press the appropriate ellipsis button to set the active style colors for a column's **Foreground Normal**, **Foreground Selected**, **Background Normal**, and **Background Selected** display mode.
Picture	Press the ellipsis button to build a picture token, or enter a picture token in the entry field provided which will replace the default column picture.

Runtime List Box

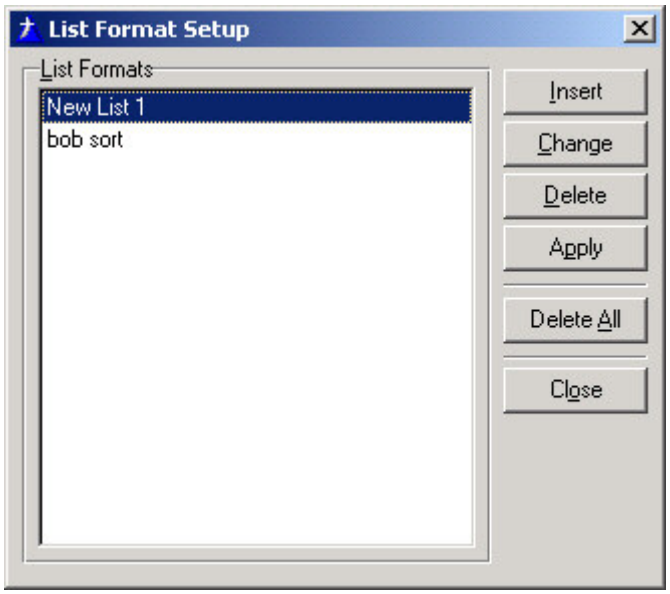
The **Runtime ListBox** style option is used to generate the appropriate runtime property syntax for a selected style and a selected list box that is not associated with the default procedure template (i.e., a hand coded list box or a third party template that populates a list box). Select the list box to associate with the current active style.

List Format Manager - Procedure Level

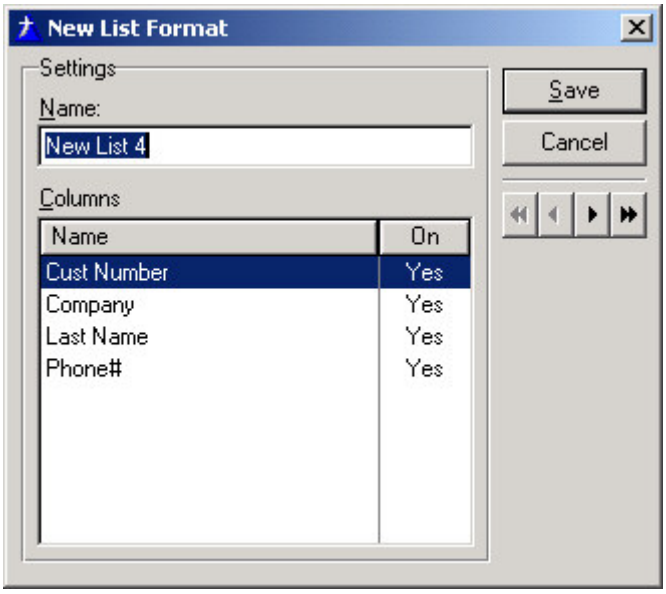
The List Format Manager dialog allows you to set the precise level of control for the runtime Format Manager. Special formatting of the list box can be controlled behind the scenes by program logic, or you can open up a special popup design interface that allows the users to create and modify unlimited list box format appearances.

The following images show an example run time Format Manager interface:

From the browse box, the user can right-click, and select a different format from the popup menu, or add a new format by selecting the List Format menu item.



The list box above shows the Format Editor's active list formats, and allows the user to add, change or delete them as needed.



The user has the ability in the above Format Editor window to designate what fields (columns) to use in the new format, and move the field values up or down in order.

Note: If you change the order that the columns are presented in the list box, each column MUST HAVE a specific column field number defined in the list box formatter (do not use the Auto option). See: Field Number

The following prompts are presented:

Popup Mode
Choose *Enable* from the drop list to allow the user to access the Format Editor interface at runtime, or *Disable* to remove this menu item from the popup.

Format Editor Interface:

Items Check Type
Select **Icons** to allow a check box icon to appear to the right of the column selection interface. Selecting **Text** will give you a "Yes" displayed when a column is included in the format string.

Items Sort By
Select **Alpha** from the drop list to display all list formats in alphabetical order. Select **Code** to sort the list formats "as is", in the order that they are entered.

Tool Tip
Select *Enable* to allow tool tips to be displayed in the Format Editor.

Different Format for each Sort Order?
Check this box if you would like to save a different list format for each active sort order defined (default and conditional). For example, a sort by account number may display a list format with the account number first, a sort by name may display a list format with the name first, etc.

Append Sort Number to Menu item
Each sort defined uses by default "Sort Order *X*" where *X* is equal to the sort number. If you are translating this text by overriding the Sort Order Menu item in the Global Properties, you can disable the sort number generation here.

Save Format on Session?
Check this box to allow users to save their current format changes when closing, and use that same format as the new default when the window is opened again.

Note:

The destination of the saved changes is defined in the List Format Global Extension. You can specify that the application create a table to store the format changes, or you can specify an existing table in your data dictionary.

Menu Editor

The **Menu Editor** dialog visually represents a Clarion MENUBAR data structure. The menu tree (on the left hand side of the dialog) appears as simplified Clarion language syntax, containing these Clarion keywords:

A MENUBAR keyword at the top.

A MENU statement or statements followed by a menu name, and a corresponding END statement.

An ITEM statement or statements followed by an item name.

Menu Editor command buttons allow you to add and delete MENUs and ITEMs. You may also move MENUs and ITEMs within the MENUBAR structure with the "up arrow" and "down arrow" buttons.

The right hand side of the dialog lets you specify the text of your MENUs and ITEMs, the equate labels used to reference the MENUs and ITEMs in executable code, and the actions that occur when the user selects an ITEM.

When using the Application Generator, each ITEM you place on a MENU or MENUBAR automatically adds an embed point to the control event handling tree in the Embedded Source dialog. This lets you easily attach functionality to your ITEMs.

Menu Editor Buttons

New Menu (SHIFT+INSERT)


This button adds a new MENU statement, its Menu Text, and its corresponding END statement. The MENU is added after the highlighted line. MENUs may be nested within other menus. MENUs may contain MENUs or ITEMs.

New Item (INSERT)


This button inserts an ITEM after the highlighted line. Note that ITEMs are used to execute commands or procedures, whereas MENUs are used to display a selection of other MENUs or ITEMs.

New Separator (CTRL+INSERT)


To add a separator bar after the currently highlighted MENU or ITEM, press the **Separator button**.

Separator bars can provide the user with a visual cue that a group of ITEMs on the menu perform related functions.

Delete Menu/Item Button (DELETE)


To delete the currently highlighted MENU, ITEM, or SEPARATOR, press the **Delete** button. If you delete a MENU statement, all ITEMs and MENUs within it, and its associated END statement are also deleted.

Move Buttons (CTRL+UP Arrow) (CTRL+Down Arrow)


To move the currently highlighted MENU, ITEM, or SEPARATOR up or down in the menu list, press the or ¯ button. When moving a MENU, all ITEMs and MENUs within it, and its associated END statement move also.

Menu Properties

Flags

Check To create an on/off toggle for a selected ITEM, set the **Check** property to True. The ITEM should have a numeric variable in the **Use** field. The variable should be declared using one of the data dialogs, or in embedded source. The **Menu Editor** adds the CHECK attribute to this ITEM.

With the CHECK attribute, when the user selects the ITEM for the first time, the ITEM is "on," the Use Variable's value is one (1), and a check mark appears beside the ITEM. When the user selects the ITEM a second time, the ITEM is "off," the Use Variable's value is zero (0) and no check mark is displayed. You should add source code to control the application's behavior depending on the state of the Use Variable.

Disable To disable a MENU or ITEM (dim the text and make it unavailable to the user), set **Disable** to True. This adds the DISABLE attribute to the MENU or ITEM statement.

The Disable box is handy when you incorporate modality into a program--that is, when one type of child window does *not* support the same commands another type does. For the type that doesn't support the command, disable the ITEM rather than omitting it. This will avoid confusing the user with menu ITEMs that disappear and reappear depending on which window is active.

Right To right justify the selected MENU on the action bar, set **Right** to True. This is available only for MENUs on the action bar. Nested MENUs (subMENUs) cannot be right justified. Checking this box displays the selected MENU, and all MENUs after the selected MENU, at the far right of the action bar.

Color

Enter a valid color equate in the **TextColor** or **BackColor** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your MENU or ITEM declaration.

General

NoMerge To tell Clarion never to merge this MENUBAR with other MENUBARs, set the **No Merge** to True. This is available only for the MENUBAR, not for MENUs or ITEMs. See the *Language Reference* for more information on the NOMERGE attribute.

Text Type the text you want displayed for this MENU or ITEM. For example, type &FILE, so the end user sees **F**ile. The ampersand within the Menu Text field signifies the character following the ampersand is the accelerator key. That is, the character is underlined, and, when the user presses the accelerator key, the action associated with the ITEM is executed.

A MENU accelerator key requires THE ALT key to take effect, whereas an ITEM accelerator key does not require the ALT key, but does require

that the ITEM be currently displayed. See Hot Key below for another method of accessing your MENUs and ITEMs.

TextFont Calls the Font dialog which lets you change the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected MENU or ITEM As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a Field Equate Label.

A Field Equate Label has a leading question mark (?), and you should make it descriptive. For example ?File shows this menu is to manipulate a file. You can refer to the MENU within executable code by its Field Equate Label.

Icon Specify the ICON you want displayed for this MENU or ITEM.

Std To specify a standard windows action for your menu ITEM, select one of the equates listed below in the **Std** field. Clarion will automatically implement the command using standard windows behavior; you do not need any other support for it in your code. The standard equate labels and their associated actions are also contained in the ..\LIBSRC\EQUATES.CLW file.

STD:PrintSetup

Printer Options Dialog.

STD:Close

Closes active window.

STD:Undo

Reverses the last editing action.

STD:Cut

Deletes selection, copies to clipboard.

STD:Copy

Copies selection to clipboard.

STD:Paste

Pastes clipboard contents at the insertion point.

STD:Clear

Deletes selection.

STD:TileWindow

Arranges child windows edge to edge.

STD:TileHorizontal

Arranges child windows edge to edge.

STD:TileVertical

Arranges child windows edge to edge.

STD:CascadeWindow

Arranges child windows so all title bars are visible.

STD:ArrangeIcons

Arranges iconized child windows.

STD:WindowList

Adds child window names to menu.

STD:Help

Opens .HLP file to the contents page.

STD:HelpIndex

Opens .HLP file to the index.

STD:HelpOnHelp

Opens Microsoft's .HLP file for the Windows Help system.

STD:HelpSearch

Opens Microsoft's Help Search utility for the .HLP file.

Position *First*

To force the selected MENU or ITEM to the first position when merging menus, set this property to True. This adds the FIRST attribute to the MENU or ITEM statement.

Position *Last*

To force the menu or item to the last position when merging menus, set this property to True. This adds the LAST attribute to the MENU or ITEM statement.

LeftOffset Specify the level of indention for the MENU or ITEM (in dialog units). This sets the parameter for the LEFT attribute.

AT (Height and Width) Control the height and width of the selected menu or item

Default If set to True, the Clarion runtime library determines the size based on the applicable font and text value.

Value When **Default** is set to False, **Value** lets you set a precise width or height in Dialog units on a window.

Help

HelpID Type either a help keyword or a context string present in a .HLP file.

If you fill in the **Help ID** for a MENU or an ITEM, when the user highlights the MENU or ITEM and presses the F1 key, the help file opens to the referenced topic.

The **Help ID** field (HLP attribute) takes a string constant specifying the key for accessing a specific topic in a Windows Help file. This may be either a Help keyword or a context string. When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press this button to open the **Input Key** dialog. Use this dialog to add the KEY attribute to your MENU or ITEM. The KEY attribute specifies a "hot" key or key combination.

A hot key is very similar to an accelerator key. A hot key or hot key combination allows the end user to immediately display a MENU, or execute the action associated with an ITEM, without mouse clicking, and without displaying the menu that contains the ITEM. Customarily, hot keys take the form of CTRL + *character*, or CTRL + SHIFT + *character*.

You may want to add the hot key combination to the menu text to signal its availability to the user. See the *Windows Design* appendix in the *User's Guide* for a list of common hot keys associated with standard windows commands.

Message Type the MSG attribute contents.

This message text displays in the status bar (if enabled) when the user highlights this MENU or ITEM.

Mode

Hide Specifies that the menu or item does not appear when the WINDOW or APPLICATION is first opened. UNHIDE must be used to display the target menu or item.

Actions (CTRL + A)

Use the Actions properties to add functionality to your menu item. Filling in these prompts causes the menu item to execute an action when the user selects the menu item.

When Pressed — From the drop down list, choose Call a Procedure, Run a Program, or No Special Action.

The procedure or program you specify executes when the user selects the menu item. The choices are:

Call a Procedure — You must specify the **Procedure Name**, and whether the procedure will **Initiate a Thread**.

Procedure Name — From the **Procedure Name** drop down list, choose an existing procedure name, or type a new procedure name. A new procedure appears as a "ToDo" item in your Application Tree.

Initiate a Thread — Optionally check the **Initiate a Thread** box. If the procedure initiates a thread, specify the Thread Stack size. Clarion uses the START function to initiate a new execution thread. You can optionally specify **Parameters**, **Requested File Action**, or both.

A MENU ITEM on an application frame toolbar that calls an MDI child procedure must initiate a thread.

Thread Stack — Accept the default value in the **Thread Stack** spin box unless you have extraordinary program requirements. To change the value, type in a new value or click on the spin box arrows.

Parameters — In the **Parameters** field, optionally type a list of variables or data structures passed to the procedure.

Return Value Variable — If the procedure called is prototyped to return a value, press the ellipsis button here to select a variable to receive the returned value.

Optionally check the **Reference Assign** check box if the variable defined is a reference variable

Requested File Action — From the drop down list, optionally select ***None, Insert, Change, Delete***, or ***Select***. The default selection is ***None***. The Global Request variable gets the selected value. The called procedure can then check the value of the Global Request variable and perform the requested file action.

Run a Program — You must specify the **Program Name**, and optionally, any parameters.

Program Name — Type the program name. The program name must be in your path or current folder, else enter the full path and executable program. Quotes are added to your entry so you don't need to enter any.

Parameters — Optionally type a list of values that are passed to the program.

No Special Action

Choose this option if you are providing your menu item's functionality with another method, such as embedded source, or an STD ID.

You may combine a procedure or program call with embedded source, but not with an STD ID.

Embeds

(CTRL + M)

Accesses the **Embedded Source** dialog for points surrounding the event handling for this menu item only.

New Structure dialog

This dialog lists the default Window and Report Structures contained in DEFAULTS.CLW. You may edit this file to modify these structures or add new window or report types.

Here is a sample of structures that are in the shipping version of DEFAULTS.CLW:

Window A standard window.

```
window  WINDOW('Caption'),AT(,,260,100),GRAY
        END
```

Window with OK & Cancel A standard window plus two button controls labeled: OK and Cancel.

```
window  WINDOW('Caption'),AT(,,185,92),GRAY
 BUTTON('OK'),AT(144,10,35,14),DEFAULT,USE(  )
 BUTTON('Cancel'),AT(144,28,36,14),USE(?CancelButton)
        END
```

System Modal Window A System Modal window.

```
window  WINDOW('Caption'),AT(,,100,100),MODAL,DOUBLE,CENTER,GRAY
        END
```

MDI Child Window An MDI Child Window.

```
window  WINDOW('Caption'),AT(,,185,92),MDI,SYSTEM,RESIZE,GRAY
        END
```

MDI Parent Frame An MDI Parent Window.

```
AppFrame  APPLICATION('Caption'),AT(,,280,100),SYSTEM,RESIZE
          END
```

Report (portrait) A blank Report in Portrait format.

```
report  REPORT,AT(1000,2000,6000,7000),THOUS,PRE(RPT),FONT('Arial',10)
        HEADER,AT(1000,1000,6000,1000)
        END
detail DETAIL
        END
        FOOTER,AT(1000,9000,6000,1000)
        END
        FORM,AT(1000,1000,6000,9000)
        END
         END
```

Report (landscape) A blank Report in Landscape format.

```
report REPORT,AT(1000,1500,9000,5000),THOUS,PRE(RPT),LANDSCAPE,FONT('Arial',10)
         HEADER,AT(1000,1000,9000,500)
         END
detail   DETAIL
         END
         FOOTER,AT(1000,6500,9000,500)
         END
         FORM,AT(1000,1000,9000,6000)
         END
       END
```

Picture Editor Dialog

This dialog lets you quickly choose and customize a picture token.

Picture tokens provide a masking format for displaying and editing variables. Picture tokens may be used as parameters of STRING, ENTRY, or STRING OPTION declarations in SCREEN structures; as a parameter of STRING statements in a REPORT structure; as a parameter of some Clarion procedures and functions; or, the parameter of STRING, CSTRING and PSTRING variable declarations. There are seven types of picture tokens:

Numeric and Currency Pictures

Scientific Notation Pictures

Date Pictures

Time Pictures

Pattern Pictures

Key-in Template Pictures

String Pictures

Pool Select a Pool picture from the drop list.

Save Press this button to save the current picture to the Picture Pool

Delete Press this button to remove the current picture from the Picture Pool

Picture The actual picture string under construction.

You can directly edit the string, or edit it by specifying options in the dynamic controls that appear in the lower part of the dialog box for individual picture token types.

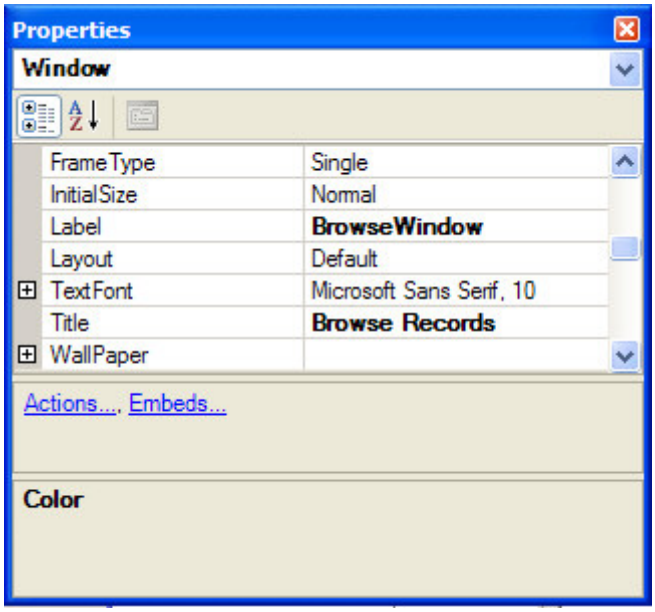
The options are self explanatory, such as **Number of Characters**, **Length**, and **Decimal Places**.

Type Choose one of the picture token types from the drop down list.

Window Designer - Properties Pad

The **Window Designer's Properties** Pad allows you to quickly specify the appearance and content of the text on each control within the window and on the window title bar. Control the font, size, style, and content of all your text, using standard entry fields and drop down lists. Every property of a selected control is listed in this pad.

Display and get focus to the **Properties** Pad by choosing **View ▸Properties,** or press the F4 key**.** Resize the **Properties** Pad by placing the cursor on the border of the box. When the cursor changes to a double headed arrow, CLICK and DRAG. Use the toolbar buttons to change the sort appearance.



As you multi-select controls in the Designer (CTRL + CLICK), any change in a property in the Properties Pad will affect all selected controls.

You can manually enter in property values in the pad, or clicking on the property name will cycle through valid choices where applicable. In many properties, there are lookup buttons that display additional selection dialogs.

Property View

The Property View allows you to set properties on one or more controls in the window.

To manually open the Property View, choose **View ▸Properties** from the IDE menu, or press the **F4** key

As you select multiple controls in the Structure (Window) Designer, the Property View shows the pertinent (common) values for the selected controls. As you change the value of any property, all selected properties are affected..

There are two views. Press the Alphabetical button  to sort all properties in alphabetical order. Press the Categorized button  to sort all properties grouped by a common category.

Window Behavior

Window Operation Mode

This option allow you to override the window settings specified in the **Window Properties** dialog. This allows an additional access point to modify the window's operation mode. **See Also:** WINDOW.

Use WINDOW Setting
Specifies no overrides to the window settings

Normal
Specifies application modal operation mode. The user must respond before moving to any other window in the application.

MDI
Specifies that the window conforms to standard MDI child behavior.

Modal
Specifies system modal operation. A system modal window takes complete control until the window is closed.

INI File Settings
Checking the **Save and Restore Window Location** specifies that a window's location is stored in the application .INI file, and will open in that position the next time the procedure is called. This is available only if you enable INI File settings in the **Global Properties** dialog.

Visual indicators on control with focus
Disable visual indicators
This check box is enabled if you have activated the Set Visual Indicators feature in the Global Properties window. Check this box to disable those settings for this window only.

Override Field Navigation key
Override Use of ENTER key instead of TAB
This check box is enabled if you have activated the Field Navigation feature in the Global Properties window. Check this box to disable those settings for this window only.

Disable Left/Right Key on List controls
Check this box to disable the left and right key from navigating to the next or previous control from any LIST control on this window. This may be necessary if a list box has horizontal scrolling active.

Enable Next Tab control selection
Check this box to navigate easily from one tab control to the next via keyboard control.

Use Exclude Global List
If you do not disable the Override Keystroke feature, you can check this box to use the list of controls that you had excluded in the Global Properties window.

Classes

Many of the ABC Procedure, Control and Extension templates provide a Classes tab or dialog. These local Classes tabs let you control the classes (and objects) your procedure uses to accomplish the template's task—that is, they override the global class settings specified in the **Global Properties** dialog. You may accept the default Application Builder Class specified in the **Global Properties** dialog (recommended), or you may specify your own or a third party class to override the default setting. Deriving your own class can give you very fine control over the procedure when the standard Application Builder Class is not precisely what you need.

Object Name
Set the object's label for the template generated code.

Use Default Application Builder Class?
Check this box to use the default Application Builder Class specified in the **Global Properties** dialog. Clear this box to use a class other than the default, and to enable the following prompts.

Use Application Builder Class?
Check this box to select a class from the **Base Class** drop-down list. The list includes all classes with the LINK attribute in \LIBSRC*.INC files. Clear this box to specify a class declared elsewhere.

Base Class If you checked the **Use Application Builder Class?** box, select a class from the drop-down list. If you cleared the **Use Application Builder Class?** box, type the class label here, and type the name of the source file that contains the class declaration in the **Include File** entry box.

Include File If you cleared the **Use Application Builder Class?** box, type the class label in the **Base Class** entry box, and type the name of the source file that contains the class declaration here.

Derive?
Check this box to derive a class based on the parent class specified above and to enable the **New Class Methods** and **New Class Properties** buttons to define any *new* properties and methods for the derived class.

This prompt is primarily to allow you to define *new* properties and methods in a derived class. To override *existing* methods, simply embed code in the corresponding method embed points.

Using **Derive?**, **New Class Methods** and **New Class Properties** makes the template generate code similar to the following:

```
MyProcess CLASS(Process)  !derive a class from the parent class

NewMethod  PROCEDURE    !prototype new class method

NewProperty BYTE      !declare new class property

     END
```

The template automatically derives from the parent class if you embed code into any of the derived method embed points, regardless of the status of this check box.

New Class Methods
Press this button to specify the *new* method prototypes to generate into the derived CLASS structure. This opens the **New Class Methods** dialog (see *New Class Methods*).

New Class Properties
Press this button to specify the new property declarations to generate into the derived CLASS structure. This opens the **New Class Properties** dialog (see *New Class Properties*).

Application Builder Class Viewer
Press this button to display classes, properties, and methods used by the ABC Templates, and the relationships between parent and derived (child) classes. This utility can help you analyze and understand the classes that the ABC Templates use.

Refresh Application Builder Class Information
Press this button if you have changed the contents of an include file (.INC) or added an include file to the \LIBSRC directory. Typically, this is needed when you install third party products that use ABC compliant classes, although you may create your own ABC compliant classes too. See *ABC Compliant Classes* for more information.The ABC Templates use information gleaned from the header files for generating embed points, loading the Application Builder Class Viewer, application conversion, etc.

Composite Class
Press these buttons to open a Classes dialog for each class used by the parent class specified above. For example, the WindowManager uses a Toolbar class, so the WindowManager's Classes dialog contains a Toolbar Class button to open a Classes dialog for its Toolbar Class.

Clarion Window Controls Toolbox

When the Structure (Window) Designer is opened, additional controls are placed in the Window Designer via the Clarion Window Controls ToolBox.

To open the Clarion Window Controls ToolBox, select **View ▸Tools**, or press **CTRL + ALT + X**.

To populate a control simply click on the control name in the toolbox, and drag it to the target location.

BOX Lets you place a BOX control on the window under construction. **See Also:** Box Control Properties .

BUTTON Lets you place a BUTTON control on the window under construction. **See Also:** Button Control Properties .

CHECK Lets you place a CHECK control on the window under construction. **See Also:** Check Box Control Properties .

COMBO Lets you place a COMBO control on the window under construction. **See Also:** Combo Box Control Properties .

ELLIPSE Lets you place an ELLIPSE control on the window under construction. **See Also:** Ellipse Control Properties .

ENTRY Lets you place an ENTRY control on the window under construction. **See Also:**Entry Box Control Properties .

GROUP Lets you place a GROUP control (group box) on the window under construction. **See Also:** Group Control Properties

IMAGE Lets you place an IMAGE control (graphic image) on the window under construction. **See Also:** Image Control Properties .

LINE Lets you place a LINE control on the window under construction. **See Also:** Line Control Properties .

LIST Lets you place a LIST control on the window under construction. **See Also:** List Control Properties .

MENUBAR Creates a MENUBAR structure in which MENU and ITEM s can be created.

OLE Lets you place an OLE control on the window under construction. **See Also:** OLE Control Properties Dialog .

OPTION Lets you place an OPTION control (OPTION structure, which appears as a group box with radio buttons) on the window under construction. **See Also:** Option Control Properties .

PANEL Lets you place a PANEL control on the window under construction. **See Also:** Panel Control Properties.

PROGRES Lets you place a PROGRESS control on the window under construction. **See Also:**

S Progress Control Properties .

PROMPT Lets you place a PROMPT control on the window under construction. **See Also:** Prompt Control Properties .

RADIO Lets you place a RADIO control on the window under construction. **See Also:** Radio Button Control Properties .

REGION Lets you place a REGION control on the window under construction. **See Also:** Region Control Properties .

SHEET Lets you place a SHEET control on the window under construction. **See Also:** Sheet Control Properties .

SPIN Lets you place a SPIN control on the window under construction. **See Also:** Spin Box Control Properties .

STRING Lets you place a STRING control on the window under construction. **See Also:** String Control Properties .

TEXT Lets you place a TEXT control on the window under construction. **See Also:** Text Box Control Properties .

TOOLBAR Lets you place a TOOLBAR control on the window under construction.

Window Designer Menu Commands

The **Structure (Window) Designer** helps you visually design Window elements--windows and controls--on screen. The **Window Designer** automatically generates and places the language structures and source code that describe these elements in your application (.APP) file or source code document. **See Also:** How to Customize Your Window

The Window Designer IDE is comprised of several elements. Each of these is described in the following links:

Control Toolbox

Property Window

Alignment Toolbar

Popup (right-click) Menu

The popup menu provides quick and easy access to a subset of the **Edit** menu commands.

Bring to Front	For overlapping controls, select this item to move the active control to the top.
Send to Back	For overlapping controls, select this item to move the active control to the back.
Format	Calls a sub-menu with all of the supported alignment functions
Show Tab Order	This option lets you visually specify the tab-stop order of the controls in the window. A small box with a number inside appears on each control, indicating the current order. CLICK on the controls to change the order to the order you wish. To cancel the tab order change, simply right-click outside of the window area and toggle this item off. Parent/Child controls are marked with the following convention: parent control.child control.grand-child control For example, a STRING that is populated on a SHEET/TAB control may show *1.0.2*. The second SHEET on the window, first TAB in that sheet, and the third control populated on that tab. This feature uses the Microsoft Forms standard, where controls are number starting at control zero (0) as the first control.
Lock Controls	This option locks all controls to their current position. A "locked" icon appears in the upper left area of the control. When a control is locked, it may not be moved or resized.
Select Template	In the Application Generator Designer, this option selects the control and any other controls associated with this template.

Actions	In the Application Generator Designer, this option selects the control's template Actions dialog, if applicable.
Embeds	In the Application Generator Designer, this option selects the control's template Embeds dialog, if applicable.
Key	Opens the **Input Key** dialog for the selected control. Establish a hot key, or key combination, that gives immediate focus to the control, or for buttons, initiates the button's action.
Alert	Opens the **Alert Keys** dialog for the selected control. Add or delete one or more keys, or key combinations, that will generate an event:ALERT when the control has focus.
Edit Text	If the control selected supports additional formatting, an extra menu item will appear here. Edit Text allows you to change the Text property of any control (if applicable).
Cut	Deletes the selected control from the work area and places it in the clipboard.
Copy	Places a copy of the selected control from the work area into the clipboard.
Paste	Pastes a control from the clipboard into the work area, at the cursor position.
Duplicate	Copies the selected control to the window. You can also duplicate by pressing the F2 Key, or CTRL+DRAG and already selected control.
Delete	Deletes the selected control. Alternatively, press the DELETE key.
Properties	Open and gives focus to the selected item's **Properties Pad**.

Edit Menu

Many IDE menu items are not directly applicable to the Structure Designer.

The following items are applicable to the Structure (Window) Designer.

Undo	Reverses the most recent editing action.
Redo	Reverses the most recent undo action.
Exit Structure Designer	Exits and saves the active structure.

View Menu

The following items are applicable to the Structure (Window) Designer.

Properties	Opens the **Properties** View. While in the Designer, all control and window properties may be inspected and modified in this window.
Toolbox	In the Report and Window Designers, this option opens the Clarion Window Controls Toolbox

Project, Build, Debug, Search and Tools Menus

These menu items contained in these menus are not directly applicable to the Structure Designer.

Window Designer Menu

The Window Designer menu is visible during any Designer session, and provides populating controls options and alignment commands for spacing and sizing the controls within the window. You may place two or more controls so that their 'edges' match up with each other. You may also spread the controls out, or make all of them the same size.

To do so, first select two or more controls. Select the first by clicking on it. Select the second and subsequent controls by pressing the CTRL key, then clicking on the second control while the shift key remains pressed.

Lasso multiple controls by CTRL+CLICK+DRAGGING to form a box around the controls.

Populate Menu:

The menu options here allow you to quickly "populate" a window with entry controls and prompts for columns in your data dictionary table. Choose the Column Option for a single column population, or Multiple to recusively use this dialog.

Format Menu: Align

Lefts	Aligns the left borders of the selected controls with the left border of the last control selected (black handles).
Centers	Aligns the center of the selected controls with the horizontal center axis of the last control selected (black handles).
Rights	Aligns the right borders of the selected controls with the right border of the last control selected (black handles).
Tops	Aligns the top borders of the selected controls with the top border of the last control selected (black handles).
Middles	Aligns the center of the selected controls with the vertical center axis of the last control selected (black handles).
Bottoms	Aligns the bottom borders of the selected controls with the bottom border of the last control selected (black handles).
To Grid	Snaps the selected controls to the nearest grid coordinate.

Make Same Size	Makes all selected controls the same height and width.
Width	Makes all selected controls the same width as the last control selected (black handles).
Size To Grid	Makes all selected controls the same height and width as the size of the grid coordinate.
Height	Makes all selected controls the same height as the last control selected (black handles).
Both	Makes all selected controls the same height and width as the last control selected (black handles).

Horizontal Spacing

Make Equal	Equalizes the horizontal spaces between the selected controls.
Increase	Increases the horizontal spaces between the selected controls.
Decrease	Decreases the horizontal spaces between the selected controls.
Remove	Removes all horizontal spaces between the selected controls.

Vertical Spacing

Make Equal	Equalizes the vertical spaces between the selected controls.
Increase	Increases the vertical spaces between the selected controls.
Decrease	Decreases the vertical spaces between the selected controls.
Remove	Removes the vertical spaces between the selected controls.

Center in Form

Horizontally	As a group (relative positions of selected controls don't change), centers the selected controls vertically within the window.
Vertically	As a group (relative positions of selected controls don't change), centers the selected controls horizontally within the window.

Order

Bring to Front	For overlapping controls, select this item to move the active control to the top.
Send to Back	For overlapping controls, select this item to move the active control to the back.

Lock Controls

Lock Controls	This option locks all controls to their current position. A "locked" icon appears in the upper left area of the control. When a control is locked, it may not be moved or resized.

Window Designer Options

The **Window Designer Options** dialog sets the default position and size values applied when auto-populating controls, or when aligning controls with the alignment tools. To access the dialog, choose **Setup ▸Window Designer Options** from the environment menu, or choose **Tools ▸Options** from the Window Designer menu. The dialog is divided into four sections or tabs: Grid, Populate Defaults, Margin Defaults, and Spread Defaults.

In order for some Window Designer settings to take effect, you may need to restart the IDE.

Grid

This tab turns grid snap on or off, and sets the starting point and offsets of the window grid. It also lets you show or hide the screen boundaries (extents) for the most common video resolutions (640x480, 800x600, 1024x768, etc.).

You can use the grid to force the boundaries of your window controls to fall only on certain x / y values (axis, latitude, longitude). By enforcing the grid axis, your controls are easier to position and align.

Snap to Grid

Set to True to turn grid snap on; set to False to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

You can also choose View ▸ Show Grid from the menu, or press the button to toggle grid snap on and off.

Horizontal

Enter the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on.

Vertical

Enter the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

Origin X

Enter the horizontal coordinate at which to begin placing the grid dots. This is the left-most position at which controls will align or auto-populate when grid snap is on.

Origin Y

Enter the vertical coordinate at which to begin placing the grid dots. This is the top-most position at which controls will align or auto-populate when grid snap is on.

All the grid values are specified in dialog unitsa unit of measure based on the current system font. See the Glossary for a complete definition.

Show Screen Extents

Check this box to show video screen boundaries within the Window Designer for the most common video resolutions. Clear the box to suppress the boundaries.

Snap on Resize

Check this box to force controls to snap to the nearest grid point grid when resizing from the right or bottom edges. This constrains a controls width and height to the grid. Resizing a control using the top and left edges always snaps to the grid.

Populate

This tab sets the default width and height for a variety of window controls. The Window Designer applies the default sizes whenever you use it to add a control to the window.

Control Type

Choose the type of control for which to set the default size.

The default sizes are specified in dialog units, a unit of measure based on the current system font. See the Glossary for a complete definition.

Default Width

Set the default width for the specified control type. A value of zero (0) specifies no width; the control expands to the size of the data it displays. See *AT* in the *Language Reference* for more information.

Use most common width already present

Check this box to specify a dynamic default based on width of any controls of the same type that are already present on the window. For example, if there are three ENTRY controls and two of the controls are 50 units wide, then 50 becomes the default width for ENTRY controls. Clear this box to always apply the Default Width value, even if other controls of the same type are present.

Default Height

Set the default height for the specified control type. A value of zero (0) specifies no height; the control expands to the size of the data it displays. See *AT* in the *Language Reference* for more information.

Use most common height already present

Check this box to specify a dynamic default based on height of any controls of the same type that are already present on the window. For example, if there are three ENTRY controls and two of the controls are 10 units wide, then 10 becomes the default height for ENTRY controls. Clear this box to always apply the Default Height value, even if other controls of the same type are present.

Margins

This tab sets the margins applied by the margin alignment tool. For more information on these alignment tools, see *Window Designer Tools*. The margin is simply the distance between the closest edges of two controls (or of a control and the window). The Window Designer applies the margins whenever you use the margin alignment tools.

Different types of controls require different margins to accommodate their unique characteristics. For example, TAB controls and GROUP controls need extra space to allow for their text.

Container Type

Choose the type of control for which to set the margins. Choose from:

Window
Set the default margins.

Group
Set the margins to apply for controls inside a GROUP control, abutting the GROUP control.

Option
Set the margins to apply for controls inside an OPTION control, abutting the OPTION control.

Tab
Set the margins to apply for controls inside a TAB control, abutting the TAB control.

Top Margin

The distance between the top edge of the selected control and the nearest horizontal edge of a bounding control or window.

Side Margins

The distance between the vertical edges of the selected control and the nearest vertical edge of a bounding control or window.

Bottom Margin

The distance between the bottom edge of the selected control and the nearest horizontal edge of a bounding control or window.

The two settings (Full and Thin) provide alternative margins applied by the margin alignment tools. For more information on these alignment tools, see *Window Designer Tools*.

The default sizes are specified in dialog units, a unit of measure based on the current system font. See the Glossary for a complete definition.

Spread

This tab sets the default spacing between auto-populated window controls and between controls positioned by the Spread Alignment tools. The Window Designer applies the default spacing when you auto-populate fields from the Fields Toolbox and when you use the Spread Alignment tools. For more information on these alignment tools, see *Window Designer Tools*.

Button Spacing

Set the default distance between the edges of a button control and the nearest control.

Other Control Spacing

Set the default distance between the edges of two adjacent controls.

Prompt to Field Spacing

Set the default distance between the right edge of a PROMPT control and the left edge of its (visually) associated (ENTRY, SPIN, TEXT, etc.) control.

The default sizes are specified in dialog units, a unit of measure based on the current system font. See the Glossary for a complete definition.

Multiple control alignments

This selection applies only to the Spread Alignment tools. For more information on these alignment tools, see *Window Designer Tools*.

Use fixed spacing
Apply the static values specified above when spacing (spreading) multiple controls.

Calculate spacing
Calculate spacing based on distance between first and last control, so there is an equal distance between each control.

Window Properties

This dialog lets you specify the appearance and functionality of your window.

Color

Enter a valid color equate in the **TextColor**, **Background**, **SelBackground**, or **SelForeground** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your window declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Custom Color dialog
For each property that requires a color attribute, there is a drop list that displays a variety of default colors to select from. However, if you need to choose a custom color, select the Custom tab, and RIGHT-CLICK on any color element in the bottom two rows of this dialog.

Design

Design options control the cosmetic settings in the Designer.

DrawGrid Set to True to display the design grid, or set to False to turn it off.

GridSize Grid values are specified in dialog units, a unit of measure based on the current system font. Enter the horizontal and vertical grid size, separated by commas. **Width** is the horizontal distance between the grid dots (x axis). This is the minimum horizontal distance you can move a control when grid snap is on. **Height** is the vertical distance between the grid dots (y axis). This is the minimum vertical distance you can move a control when grid snap is on.

Locked "Freezes" all the controls on the window so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

SnapToGrid Set to True to turn grid snap on; set to False to turn it off. Grid snap displays a dot grid of valid positioning coordinates and forces the upper left corner of new controls to align with the dot grid. The end user does not see the grid at run time; it is a design tool only.

Extra

Drop ID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Icon To associate an icon with the window, allowing it to be minimized, specify an icon in this field. You may type in a file name or an EQUATE. You may also press the ellipsis button (...), then select an icon file name using the standard **Open File** dialog. The file name or equate you specify becomes the parameter for the ICON attribute.

You should always specify an icon for an application window, and for an MDI child window. Specifying an icon name automatically places a minimize button on the caption bar of your application or MDI child window.

Palette Use the PALETTE attribute to specify maximum color depth. The PALETTE attribute specifies how many colors you want this window to use when it is the foreground window. For example, 24-bit color would be 16777215. The number you specify becomes the parameter for the PALETTE attribute. Leave this field blank to specify the default for the end user's system.

Statusbar-Status To provide a message bar at the bottom of your window, set the **Status** property to True. This adds the STATUS attribute to the window.

A status bar in an application window is an excellent way to increase user feedback in your application. Clarion makes it simple to post messages on the status bar advising the user of what your application is doing as it does it. Increasing user feedback makes the user feel more in control. This allows the user to feel more confident and be more efficient when using your application.

Widths To set the width of the status bar zone(s), type a value or list of values in the **Status Widths** field. You must also check the **Status Bar** box in the top part of the dialog to display a status bar.

The values you enter in this field fill the STATUS attribute parameters. For example, To create a two part status bar with a width of 100 in the right status section and default width in the left section enter the values '-1,100'

The zones are the areas within the status bar marked off by the 3D shaded boxes. The first zone on the left, by default, displays MSG attribute text. This is useful for specifying short help instructions or other information to the user. If your application has only one zone for the status bar, you may omit this field. For more than one zone, enter a series of comma separated values. The default measurement unit is dialog units.

You may set a minimum value for a zone width by typing a negative number. This creates a zone with a minimum width, but is expandable by resizing the window. Use the runtime property assignment syntax to place text in any zone. To place a string constant in the second zone, for example:

{PROP:StatusText,2} = 'Record will be Added'

A multi-zone status bar can give your application a professional look. You may display help text in zone one, and when editing a record, the current record number in zone two, for example.

Timer To have the window receive Timer Event messages from Windows, fill in the **Timer** field. Specify the timer interval in hundredths of seconds. The value you specify becomes the parameter for the TIMER attribute.

For example, if you specify 100 in the field, the window will automatically receive an EVENT:Timer once every second (100/100's seconds). This might be appropriate for adding a clock to a status bar.

General

Frame Type To choose the frame for your window, pick a selection from the **Frame Type** drop-down list. The frame defines the borders of the window. Choose from:

Single - a single pixel frame which the user cannot resize. Most suitable for dialog boxes.

Double - a thick frame, which the user cannot resize. Use this type frame for a system modal window with no caption bar, or for a modal dialog box with a caption bar. This adds the DOUBLE attribute to the window.

Resizeable - a thick frame, which the user can resize. Choose this for application and MDI child windows. This adds the RESIZE attribute to the window.

Initial Size Sets the initial state of your window. Choose from:

Normal:- displays the window at the default size which either you specifically set, or Windows sets if you don't.

Maximized: - the window fills the desktop, if an application window, or the window frame, if an MDI child window. This adds the MAXIMIZE attribute to the window.

Iconized: - the window appears in iconized state--as a 32 by 32 pixel window at the bottom of the desktop, for an application window, or at the inside bottom of the application frame, for an MDI child window. This adds the ICONIZE attribute to the window.

Label This names the specific WINDOW in the source code. The label may contain upper or lower case letters, numerals, the underscore character or a colon. Space characters are forbidden. The first character must be a letter or the underscore character. Clarion reserved words may not serve as labels.

Layout Indicates the orientation of window controls and field sequence.

Left to Right maintains the original layout specified in the Window Designer.

Default field navigation moves from left to right.

Right to Left essentially "flips" the window controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left.

The setting in the Application Frame will cascade its setting to all child window that have the **Default** setting active.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Title To specify caption bar text, type a string constant in the **Title** field. The caption bar holds the name of the window.

You may dynamically alter the caption bar text at run-time. See PROP:Text in the Language Reference.

Wallpaper-BackgroundImage To provide a background image for the window's client area, specify an image filename. Type the filename or press the ellipsis button (...) to select a file. See WALLPAPER in the *Language Reference*.

Mode Specify how the window displays the background image. Choose from:

Stretched The image expands to fill the entire client area.

Centered The image displays at its default size and is centered in the window's client area.

Tiled The image displays at its default size and is repeated so it fills the entire client area.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Misc

IsSuppress Transparency

Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Options:

AutoDisplay To add the AUTO attribute to your window, set the **AutoDisplay** property to True. This automatically updates the contents of all controls on screen through each pass of the ACCEPT loop.

Docking The options in the **Dock** property group are enabled only if you have set the **Toolbox** property to True. Click on a location to specify the area that the window is allowed to be docked to within the application frame.

The **InitialState** property drop list is used to specify where you would like the window to first appear.

EntryPattern To enable support for an entry mask for controls in the window, set **EntryPattern** to True. This lets you specify key-in entry patterns for the fields you choose, and adds the MASK attribute to the window.

Gray To provide the gray window background, chiseled control look for your application, set the **Gray** property to True. This is clearly a style consideration, but will go a long way in giving your application a professional look. This adds the GRAY attribute to the window.

The gray background is not visible when you design your window with the **Window Designer**. It is, however, visible in test mode, and when your application runs.

Immediate To generate a message event each time the end user moves or resizes the window, set the **Immediate** property to True. This adds the IMM attribute to the window. You are responsible for the code that executes upon notification of the event.

MaximizeBox To place a maximize button in your window, set the **MaximizeBox** property to True. In general, you should place a maximize button only on application windows and MDI child document windows. This adds the MAX attribute to the window.

MDIChild To add the MDI attribute to your window, set the **MDIChild** property to True. An MDI child window cannot move outside the main application window. A typical use of an MDI window might be to present a different arrangement of the data in your application's database.

ModalWindow To specify a system modal window, set the **ModalWindow** property to True. A system modal window prevents all other tasks--even in applications other than your own--from executing until the window is closed. This adds the MODAL attribute to the window.

SystemMenu To place a system menu in your window, set the **SystemMenu** property to True. Most windows should have a system menu. For users on a system without a mouse, the system menu provides the only means of minimizing, maximizing or re-sizing the window. This adds the SYSTEM attribute to the window.

Even if you plan that the window should NOT have a system menu when the application is complete, it's good practice to place a system menu on your application while it's under development. By DOUBLE-CLICKING the system menu, or choosing Close, you can close your application should your normal exit procedure fail.

Toolbox To add the TOOLBOX attribute to your window, set the **Toolbox** property to True. This makes your window always stay on top. **See Also:** DOCK

Position

Lets you set the location and size of a window.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the size visually by dragging with the mouse in the **Window Designer**.

To set the location of the window's **Top Left Corner**, choose from the following options for the **X** (horizontal) and **Y** (vertical) coordinates.

Center Centers an APPLICATION window on the screen. Centers child windows on their respective parents.

Default The Windows operating system determines the initial position of the window. By default, Windows position windows in a cascading sequence from top-left to bottom-right.

Value Lets you set a precise coordinate in Dialog units. Generally, the coordinate is relative to the top left corner of the screen for APPLICATION (first or main) windows and relative to the top left corner of the APPLICATION window for all subsequent (child) windows.

To give your application the "standard" look of other Windows applications, where possible specify the *Default* setting for any windows with resizable frames.

To set the window's size, choose from the following options for the **Width** and **Height**.

Default The Windows operating system determines the initial size of the window.

Value Lets you set a precise width or height in Dialog units. Dialog units provide a relative screen measure based on the system font character size.

Sizing all windows and controls in dialog units lets you design a screen at one resolution, and expect it to look similar at another--in theory. In practice, there can be differences, especially when you display bitmaps in Image controls. Therefore, test your applications in the popular Windows resolutions. The most popular are 640 x 480, 800 x 600, and 1024 x 768 pixels.

Scrollbars

These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the window.

Horizontal To add a horizontal scroll bar to your window, set the **Horizontal** property to True. Scroll bars only appear when something inside the window--a control--is bigger than the window.

Vertical To add a vertical scroll bar to your window, set the **Vertical** property to True.

Quick Links

Actions Accesses the Actions dialog for this procedure.

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this procedure.

Class Tab Properties

Classes Tab

Many of the ABC Procedure, Control and Extension templates provide a Classes tab or dialog. These local Classes tabs let you control the classes (and objects) your procedure uses to accomplish the template's task—that is, they override the global class settings specified in the **Global Properties** dialog. You may accept the default Application Builder Class specified in the **Global Properties** dialog (recommended), or you may specify your own or a third party class to override the default setting. Deriving your own class can give you very fine control over the procedure when the standard Application Builder Class is not precisely what you need.

Object Name
Set the object's label for the template generated code.

Use Default Application Builder Class?
Check this box to use the default Application Builder Class specified in the **Global Properties** dialog. Clear this box to use a class other than the default, and to enable the following prompts.

Use Application Builder Class?
Check this box to select a class from the **Base Class** drop-down list. The list includes all classes with the LINK attribute in \LIBSRC*.INC files. Clear this box to specify a class declared elsewhere.

Base Class
If you checked the **Use Application Builder Class?** box, select a class from the drop-down list. If you cleared the **Use Application Builder Class?** box, type the class label here, and type the name of the source file that contains the class declaration in the **Include File** entry box.

Include File
If you cleared the **Use Application Builder Class?** box, type the class label in the **Base Class** entry box, and type the name of the source file that contains the class declaration here.

Derive?
Check this box to derive a class based on the parent class specified above and to enable the **New Class Methods** and **New Class Properties** buttons to define any *new* properties and methods for the derived class.

This prompt is primarily to allow you to define *new* properties and methods in a derived class. To override *existing* methods, simply embed code in the corresponding method embed points.

Using **Derive?**, **New Class Methods** and **New Class Properties** makes the template generate code similar to the following:

```
MyProcess CLASS(Process)  !derive a class from the parent class
NewMethod  PROCEDURE    !prototype new class method
NewProperty BYTE      !declare new class property
     END
```

The template automatically derives from the parent class if you embed code into any of the derived method embed points, regardless of the status of this check box.

New Class Methods
Press this button to specify the *new* method prototypes to generate into the derived CLASS structure. This opens the **New Class Methods** dialog (see *New Class Methods*).

New Class Properties
Press this button to specify the new property declarations to generate into the derived CLASS structure. This opens the **New Class Properties** dialog (see *New Class Properties*).

Application Builder Class Viewer
Press this button to display classes, properties, and methods used by the ABC Templates, and the relationships between parent and derived (child) classes. This utility can help you analyze and understand the classes that the ABC Templates use.

Refresh Application Builder Class Information
Press this button if you have changed the contents of an include file (.INC) or added an include file to the \LIBSRC directory. Typically, this is needed when you install third party products that use ABC compliant classes, although you may create your own ABC compliant classes too. See *ABC Compliant Classes* for more information.The ABC Templates use information gleaned from the header files for generating embed points, loading the Application Builder Class Viewer, application conversion, etc.

Composite Class
Press these buttons to open a Classes dialog for each class used by the parent class specified above. For example, the WindowManager uses a Toolbar class, so the WindowManager's Classes dialog contains a Toolbar Class button to open a Classes dialog for its Toolbar Class.

New Class Properties

Press the **Insert** button to add a new property declaration.

Property Name Type the property label.

Property Type Select a simple data type from the list or select *Other* to enable the **Other Data Type** field.

Other Data Type Type the label of a user defined complex data type (such as the label of a GROUP, QUEUE or CLASS), or type a valid entity data type (such as FILE, VIEW, or WINDOW).

Is a Reference Check this box to declare a reference variable. You must use a reference variable for entity data types and for any complex data type not valid within a GROUP. You may use a reference variable for any other data type. See *GROUP* and *Reference Variables* in the *Language Reference*.

Size Specify the length of the field in bytes.

Dimensions To declare the field as an array, and to specify the array dimensions, specify a size for up to four dimensions. Total array size may not exceed 65,520 bytes. See the *Language Reference* for more information on dimensioned variables and arrays.

New Class Methods

New Class Methods	Press the **Insert** button to add the new method prototype and the method's associated embed points.
New Method Name	Type the method label.
New Method Prototype	Type the method parameter list and return data type. If the method takes no parameters but has a return value, type parentheses and a comma before the return data type. Do not type "PROCEDURE" or "FUNCTION" because the template generates the PROCEDURE statement for you.
Data Embed	Press this button to use the Text Editor to implement the method's data section.
Code Embed	Press this button to use the Text Editor to implement the method's code section.

Application Builder Class Viewer

This utility display classes, properties, and methods used by the ABC Templates, and the relationships between parent and derived (child) classes. This utility can help you analyze and understand the classes that the ABC Templates use.

Select the appropriate tab for the type of display.

Methods and properties are color-coded to let you know if they are PRIVATE, PROTECTED, or VIRTUAL.

To locate a specific Methods and property, press the **Find** button.

Optionally, you can filter the list by clearing the **Show Private** and/or the **Show Protected** checkboxes.

The class viewer is 32-bit. It is called from the IDE 32-bit server and will not close automatically on exit from the IDE. The viewer window must be closed explicitly.

This viewer only lists ABC Compliant Classes. See *ABC Compliant Classes* for more information.

Find In ABC classes

Find What Type in the search string.

Direction. Set the direction option you want to control the search.

Find Press this button to start the search.

Text Editor

Text Editor Menu Commands

The Text (Source) Editor is a full function programmer's editor featuring Multiple Document Interface support, auto-indent, search-and-replace, and color-coded syntax highlighting.

Click here to jump to the Text Editor Configuration Options.

Related Topics: Redirection File

Use the floating Fields toolbox to insert fully qualified variable and field names at the insertion point!.

You can also get help for a keyword within a document window by placing the insertion point on the keyword, and pressing the F1 key. This lets you quickly look up help for a Clarion language statement, function or attribute.

Note that some of the IDE menu commands, most notably on the Project and Tools menus, do not specifically reference Text Editor functions. Because the Project System and the Registries are always active, their menu commands are always available.

Popup (right-click) Menu

The popup menu provides quick and easy access to a subset of the **Edit** menu commands.

If you right-click inside any opened document, the following ooptions are available:

Format	Open a submenu of common formatting options, including case toggle, commenting, comment regions, indentation, etc.
Cut	Deletes the selected text from the document and places it in the clipboard.
Copy	Places a copy of the selected text from the document into the clipboard.
Paste	Pastes text from the clipboard into the active document, at the insertion point.
Delete	Becomes active when anything is selected, and deletes the selection.
Save	Saves the active source code document.
Save As	Saves the active source code document under a new name which you specify.
Close	Closes the active source code document.
File Mode	Toggle the editor’s highlighting options, if desired.

File Options	Opens a subset dialog of the Text Editor configuration options.

In addition, in the new text editor interface there is support for multiple documents that can be navigated via the tab controls. Right-clicking on any of these tabs opens a slightly different pop-up menu:

Close	Closes the active source code document.
Close All Documents	Closes all opened documents.
Close all but this	Closes all opened documents except for the one currently selected.
Save	Saves the active source code document.
Save As	Allows you to save the active source code document to a new name.
Save All (Ctrl + Shift + S)	The term "buffers" refers to all opened documents. Selecting this item saves all opened documents.
Copy file/path name	This selection copied the full path and name of the active document to the Windows clipboard.
Open Containing Folder	Open's the IDE file dialog window with the folder in which the active document is located.

When a file with an extension of TPL or TPW is detected, the Template Editor is activated, and these additional items are also available:

Search Symbol Search Symbol Declaration	You can now search for all occurences of a givel template symbol, or where it is originally declared. Simply position your cursor over the target symbol, right-click, and select one of 6 options. You can search the current file, the current buffer, or all files in the current redirection path

Search and Replace

Search single or multiple documents using this dialog, and optionally replace the target search string.

This dialog serves both Find and Replace options. Clicking on the Replace toolbar button adds an additional **Replace with:** prompt.

Find what:
Enter a valid search string, or select a previous search string from the drop list.

Replace with:
Enter a valid string to replace the target search string.

Look in:
The **Look in** option sets the range of the current search. You can search in the active open document only, the selected elements of an active document, all open documents, the entire active project or entire solution (where multiple projects are possible). Press the ellipsis button to select an external folder to search.

Include sub-folders
If you are searching in a target folder, check this box to allow the search to cascade into any sub-folders.

Look at these file types:
Enabled if your **Look in** option is a folder. Enter the file type extensions to search into. Use a semicolon to separate multiple extensions.

Match case and Match whole word
To limit your search click on the check box in front of **Match Whole word** only or **Match Case** before clicking on the Find Next button.

Use
Select the type of search to execute. *Standard* search looks for the exact string in the Find dialog. *Regular Expressions* searches based on the result of a valid expression entered. Basically, a regular expression is a pattern describing a certain amount of text. Their name comes from the mathematical theory on which they are based. Your regular expressions require the C# syntax.

Wildcards allows the use of wildcard characters in the search string.

IDE Code Regions

!region <labelName>

<some source text>

!endregion

!region	Begins a foldable code region
labelName	An optional text description that displays when the code region is folded
!endregion	Terminates a foldable code region

A code region lets you specify a block of code that you can expand or collapse when viewing it in the Clarion# IDE.

Example:

Here is how a code region looks in the IDE:

```
8         MEMBER('')
9     !region NAMESPACE and USING
10        NAMESPACE('Threads')
11        USING('System')
12        USING('System.Drawing')
13        USING('System.Windows.Forms')
14    !endregion
15        !!! <summary>
16        !!! Description of ${ClassName}.
17        !!! </summary>
```

Region Expanded

```
8         MEMBER('')
9     NAMESPACE and USING
15        !!! <summary>
16        !!! Description of ${ClassName}.
17        !!! </summary>
```

Region Folded

Text Editor Options Dialog

To personalize your editing environment, customize the appearance, cursor behavior and other items use the **Text Editor Options** dialog.

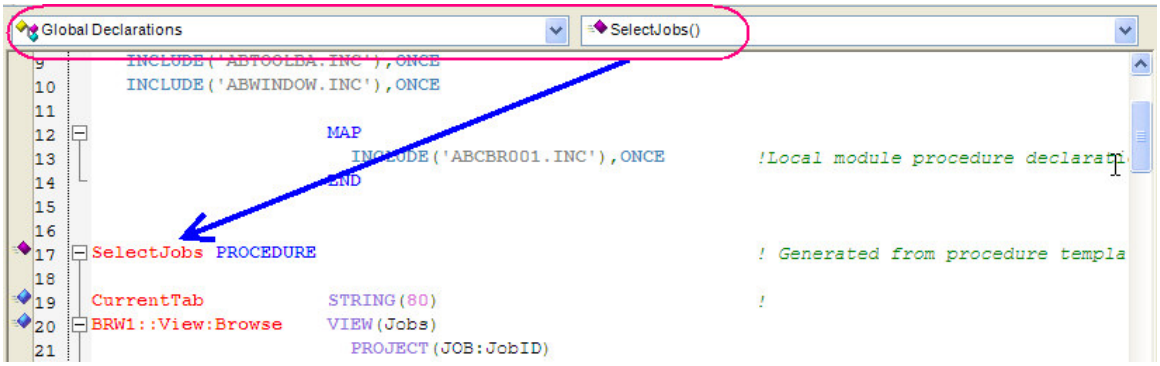
To view the dialog, choose **Tools ▸Options**. From the main IDE menu. Select the *Text Editor* tree item, which reveals the following set of sub-options.

General

Enable doublebuffering
On some machine configurations, the drawing of multi-colored text and other items can sometime cause flicker, due to the multiple passes that must be performed. Check this box to enable double buffering, which draws each line of text into an off-screen buffer before displaying it visually. This can have an effect of slower loading in some older machines, but doublebuffering in most cases is recommended.

Enable folding
Check this box to enable the folding of all data and code structures. This feature is handy for consolidating complex code sections, but also has a disadvantage of losing readability with more complex nested structures. Folding in most cases is recommended.

Show Quick ClassBrowser Panel
Check this box to display a special panel that allows you to quickly jump to selected areas of the text file.



The drop list on the left lists all data structures, like files and queues. The drop list on the right stores class structures used in this text file

Font
The Font settings allow you to control the appearance of text displayed in the Text Editor. A sample window is provided to view the font and other options that you have selected.

Markers and Rulers

The Markers and Rulers section controls the majority of the Text Editor's visual elements:

Show Horizontal Ruler
This option places a ruler at the top of the Editor window.

Show column ruler
Places a vertical line marker at whatever column you specify.

Line Marker Style
This option controls how your active line is displayed. Select Full Row for a special color indicator on the entire line.

Show line numbers
Activate this option to show line numbers on the left side of the editor window.

Underline errors
Activating this option will trigger an underlining of any errors produced by the compiler.

Highlight matching bracket
This option locates and highlights matching brackets whenever any bracket is active. This option is valid for ( ) [ ] { }

Show invalid lines
Activate this option to show invalid lines detected by the type of configuration that you have loaded. Using the "Clarion" highlighting scheme (the default), no lines are considered invalid.

Show EOL markers
Activate this option to show the "end of line" markers. These markers are identified by the special paragraph mark "¶"

Show spaces
Activate this option to show space markers. These markers are identified by standard "dot" markers.

Show tabs
Activate this option to show tab markers. These markers are identified by the special " **»** " character.

Behavior

The Behavior dialog window controls a wide spectrum of options that greatly affects your productivity in the Text editor.

Tab size
Enter the tab size in character size increments. Pressing the TAB key moves the caret (the place where something is to be inserted in a line) the specified number of characters.

Indentation size
Enter the indentation length in character size increments. Pressing the ENTER key indents the caret the specified number of characters.

Indentation (CTRL + I)

indentation functionality is divided into 2 essential parts:

- Indentation of the existing text (Ctrl + I) or formatting after pressing the Enter key.
- Cursor positioning on a new line.

There are three modes of indentation available in the Text Editor:

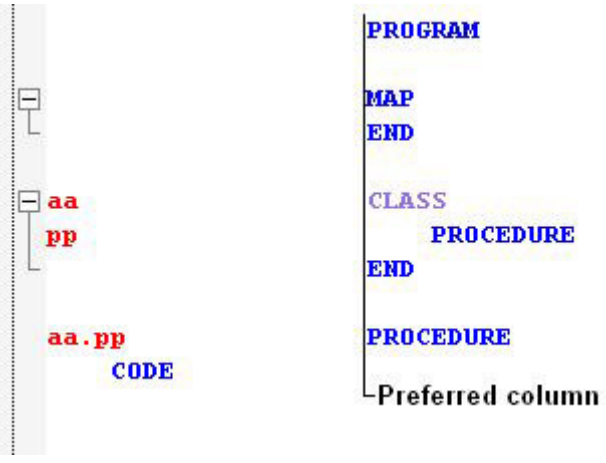
None
Set this to turn off all indentation effects. Existing text is not indented neither automatically nor by the CTRL+I hot key. Cursor is always placed to the first column on a new line.

Automatic
With Indentation set to Automatic ,the Text Editor keeps a running indent. When you press the Enter key, spaces and tabs are inserted to line up the insert point under the start of the previous line. In this mode indentation is set to the same value as on the previous line. For example, if the text on the previous line starts from the column 8, indented existing text will start from the column 8 (or the cursor on a new line will be placed to the column 8). If you try to indent the entire text in the document (CTRL+A, CTRL+I), all lines will start from column 1.

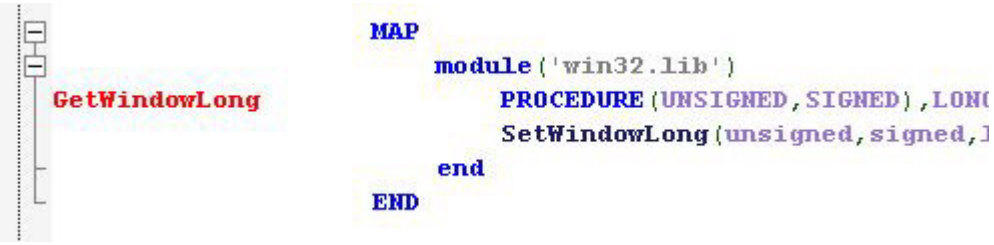
Smart
After a keyword statement, the next line is indented by the **tab size** set (see above). After certain keywords (break, return etc.) the next line is "non-indented".

Other rules for Smart include

- The **Preferred column** number (found in the Clarion Specific Options section) identifies a column where a keyword will be placed if it is not located already in some code block. In other words, if the keyword should not be indented relative to the parent it will be placed in the preferred column. The default value is column 21.
- Statements are indented from the CODE keyword position (default = on). CODE, DATA and INCLUDE (and USING and INLINE in Clarion#) keywords have one indent.
- END is indented to the same column as a keyword in the line that opens a code block:
- Keywords inside a code block have one indent from a parent keyword. For example, in the *pp* PROCEDURE:



- If a procedure inside a MAP (MODULE) is written without the PROCEDURE keyword, it will be indented:



- Any expression that ends with a colon ":" is treated as a label in the CODE section (default = off, where colons are treated as an actual procedure call)
- A line is indented after the Enter key is pressed at the end of the line (The default is on).
- You can automatically indent several lines of pasted text (The default is off).
- Source comments may also be indented (The default is off).
- If previous line has line continuation character '|' the next line will have one indent from the previous one.
- In the code definition section (prior to the CODE statement) the cursor is placed on column 1 on a new line. In the CODE section, the cursor is indented according to written statements.

Finally, if you are transferring code from another developer, and there are different **Tab Size** or **Convert Tabs to Spaces** options or the code is formatted manually, you may see the erratic behavior when using Smart indentation mode. This is due to the line(s) formatted according to your settings but all other lines remaining unchanged. To solve this issue you have several options:

1) Use the same tab size and indentation mode settings as the other developer.
2) Reformat the code before changing it (Ctrl+A, Ctrl+I).
3) Uncheck the **Enable entered line formatting** option so that the line will not be reformatted after pressing the Enter key.
4) Change indentation mode to *None* or *Automatic.*

See the Clarion Specific options dialog to control the options described above.

Convert Tabs to Spaces
When this option is active, all tab characters are auto converted to space characters when the document is opened.

Can move caret behind EOL
Checking this option allows you to move the caret marker beyond the EOL marker.

Automatic template insertion
This feature is not applicable to Clarion 7 at this time.

Auto insert curly braces
This feature is not applicable to Clarion 7 at this time.

Hide mouse cursor while typing
Check this option to hide the mouse cursor while typing. The mouse cursor will reappear when the mouse is subsequently moved.

Cut or Copy entire line when nothing is selected.
Activating (checking) this option will automatically move the entire line to the clipboard during any Cut or Copy action.

Mousewheel direction
This option is used to set the default scrolling direction of the mouse wheel.

Clarion specific options - General

The options contained in this section are unique to the Clarion IDE. The Smart Indentation options located in the Text Editor behavior section can be customized in more detail as follows:

Preferred column
Identifies the preferred column position of indentation, if the line to be indented is not based on a Parent keyword (e.g., CODE, IF, CASE, etc.).

Enable entered line formatting
Indents a line after the Enter key is pressed at the end of the line.

Indent statements from CODE
Allows indentation to be positioned relative to the location of the CODE statement.

Treat statement ends with colon as label
This option tells the editor to treat expressions that ends with a colon (**:**) as a statement label in the CODE section. If this option is off, expressions ending with a colon are treated as a procedure call. You need to have the caret position at the beginning of the target statement, and then press Enter. With option on, statement is moved to column one on the next line. With this option off, the statement is indented.

Indent comments
Indicates that any commented text will be treated as other non-commented text based on the current settings. For example, if the "pasting" option is active below, comments will be indented when pasting. If this option is off, comments will not be indented when pasting.

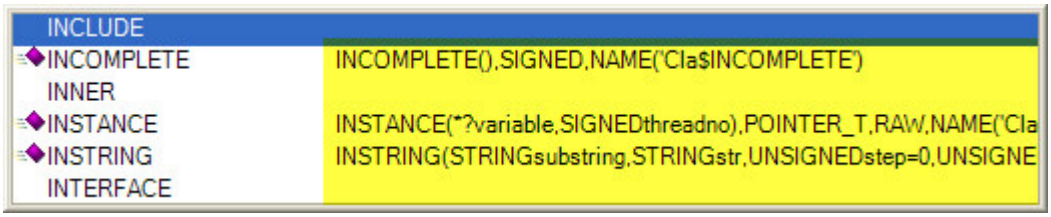
Format text after pasting several lines
Auto format (e.g., indent according to your settings) pasted text if several lines are pasted.

Clarion specific options - Clarion for Windows

Code Completion Options

Add code snippets to code completion and word completion lists
Check this option to place the IDE Code Snippets into the code and word completion lists.

Display declaration information in code completion list
Check this option to display the full declaration of the code statement, highlighted below:



There are also two resizing options available for the code completion window. Specify the **Number of rows in the completion list** to show, and also you have the option to **Remember completion list width**.

Complete options:

Keywords, Clarion attributes, and built-in data
Designate how to complete, either in upper case or lower case.

Other names
Designate how to complete all other names that are not keywords, attributes, or built-ins. Select from *As declared*. *upper case* or *lower case*.

Auto Insertion options:

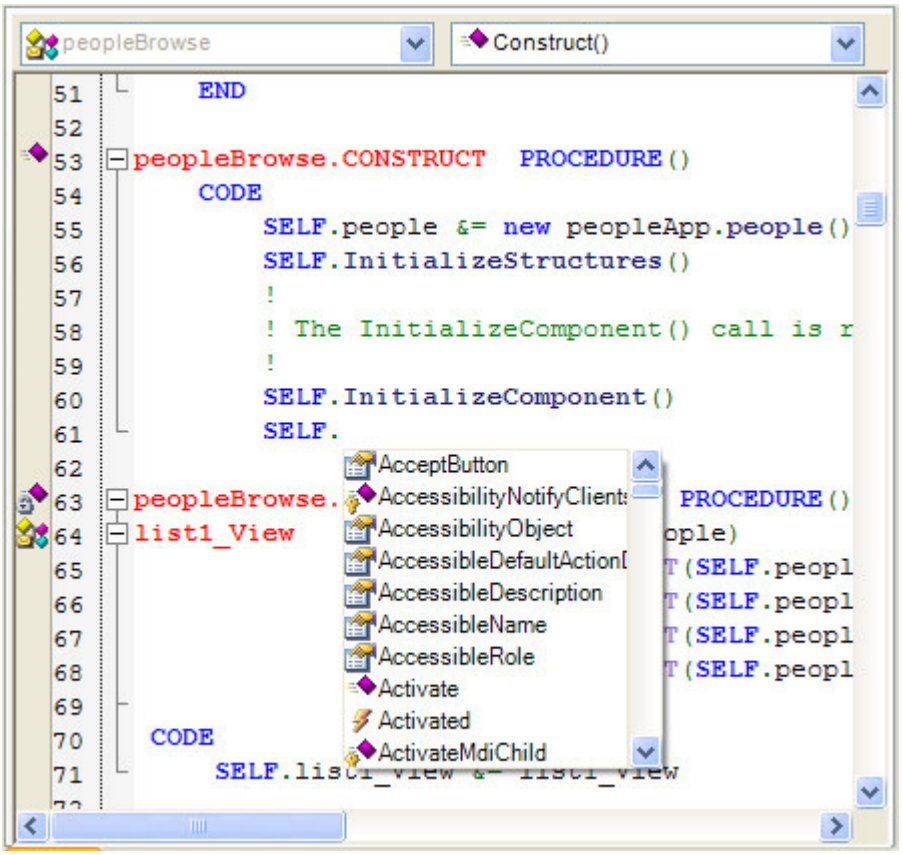
Insert line continuation character after enter is pressed
When this option is active, a line continuation character will automatically be added on enter if the last character on the line does not meet the criteria entered below.

Exclude Characters
By default, line continuation characters are not inserted if the last character in the line is a letter, digit, or one of the characters entered as shown.

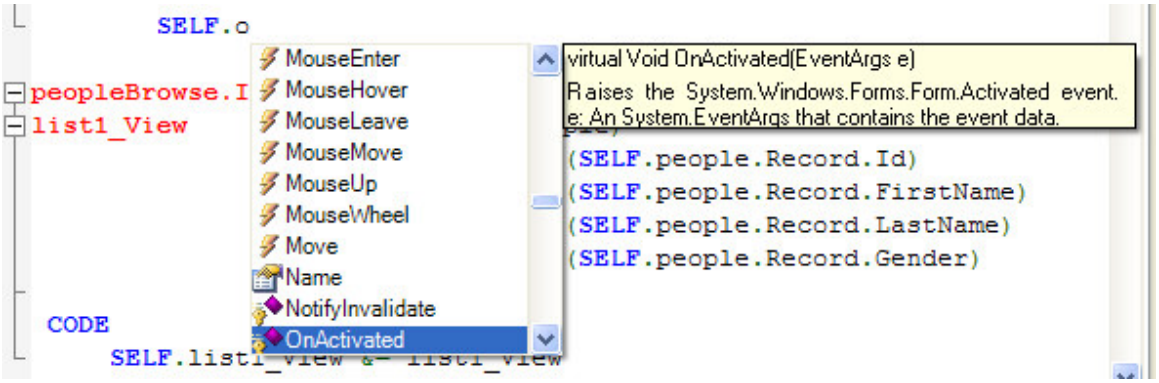
Code Completion

Clarion provides code completion as you type helping you to quickly find and enter an object's method, property or event.

Typing the dot character "." after an object's name will automatically bring up a list of possible methods, properties and events that are available for that object.



Typing the first few characters of the method, property or event will select the matching list item. Pressing the **Tab** key will complete the code by inserting the currently selected list item into the source code.



You can toggle the Code Completion feature here, on the IDE **Edit** Menu, IDE toolbar, or by hot key.

In addition, after certain keywords (like NEW and USING), pressing **CTRL + SPACE** will also display the code completion drop list. Enter **ALT** + **RightArrow** to see only the identifiers that start with typed expression.

Enable Code Completion
Check this box to activate the code completion feature.

Pre-select recently used members
Check this box to allocate a memory cache of recently used items. This will allow for quicker parsing in large projects. Set the number of items in the **Save** option, and press the **Clear Cache** button to remove all items from memory.

Show tooltip when moving mouse over expression
Check this box to activate the assisted tooltip when moving your mouse over any valid expression. Check the **Only in debug mode** box to only enable in Debug mode.

Narrow down completion list on typing
This option causes the typed word to act as a filtered locator for the code completion list.

Add new line after enter in case of fully typed word
If you press the ENTER key, and the word selected in the completion list equals the word in the typed text, a new line will be inserted. Otherwise, if the option is disabled, the ENTER key just closes the completion list.

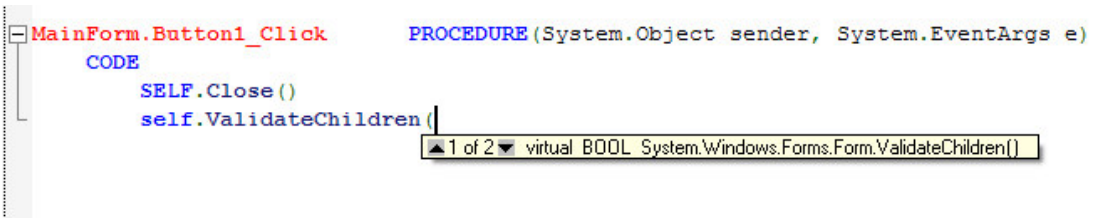
Complete word after 'insertion key' (space, dot, bracket, etc.)
If enabled and you press a dot after partially typed word, the word will be completed. For example, you type "sys" and a "System" is selected in the completion list, after pressing a dot you will have "System.". If the option is disabled, you will have "sys.". "Insertion key" is actually any character except letters, digits and underscore (and colon in Clarion).

Language Dependant Options

Show completion list after a character is typed
When active, the code completion window is automatically displayed as you type in any character that does not exist on column one (1).

Trigger code completion after keywords
Shows code completion after some of the language keywords. For example, in C# after you press SPACE after the "new", "as", "is", "override", etc. keywords, you will see the code completion window with suggestions (this works also if you enter an opening "(" after some keywords, like "for", "foreach", etc.). In Clarion/Clarion# there is only one keyword (DO) that triggers a completion list, and this displays a list of routines that can be called.

Show tooltip when writing method calls
When active, as the parameters of any method call is entered (by opening parenthesis), a tool tip pops up that offers the possible method parameters, if applicable. When this is active, you also have the option to **Re-open tooltip with better overload when pressing comma.** This checkbox reopens the tooltip with a more restricted set of choices, based on the parameters already entered. If the method is overloaded you will see all variants in the tooltip. But if you start writing the arguments of the method call, the number of possible variants is reduced so at the next comma the tooltip should be updated to show only overloads that receive typed arguments. This functionality should work in C# projects but at this time is not supported in Clarion.



XML Options

The Text Editor has built in support for XML (Extensible Markup Language). You can edit whole XML documents, or embed XML documentation in any Clarion project. The mechanics of XML Documentation is found in the Code Completion help topic.

Show attributes when folded

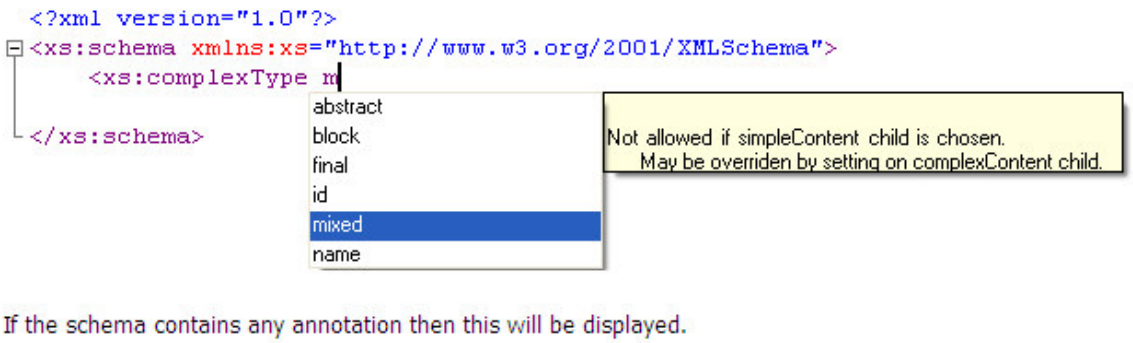
XML tags attributes can be folded. Check this option to display the XML attributes in the folded state. Example:

```
<name attribute="value">content</name>
```

If this option is active, the folded text will include the "attribute=value" text. If unchecked, only the <name> tag would be visible.

Show schema annotation

XML Completion is built into the Clarion IDE. When this option is active, you will see a display similar to the following:



If the schema contains any annotation then this will be displayed.

XML Schema

The list of active schemas used in the code completion of any XML document are listed here. To add new schemas, press the **Add** button. Only user-defined schemas can be removed from this list (built-ins are disabled)

The File Extensions list contains file extensions that are associated with the XML editing capabilities. If you need to add additional extensions, add them here.

Highlighting

The Clarion Text Editor provides several built-in highlighting schemes for a wide variety of languages. Each scheme is activated by the designated file extension.

To modify any existing scheme, highlight the desired scheme and press the **Copy to user-defined** button. From there, highlight in the right pane the scheme just copied, and press the **Modify** button. In the subsequent dialog, you can modify file extensions, keywords, color schemes, and rule sets.

Window and Report Control Properties

Box Control Properties

The box control lets you place a square or rectangle in your window or report. You may fill it with a color, specify a border color, and specify that the borders be rounded. The box control cannot receive focus, nor can it generate events.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in **Border Color** or **Fill Color** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See also **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Round	Lets you specify that the box control should have rounded edges.
Extend	**Valid in REPORT only.** Enter a valid string of attributes that are assigned to a designated REPORT control for a given document type. For more information, see EXTEND.

General

Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window

Designer.
Right to Left essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left.
The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Line Width Set the thickness of the box's border by entering a point value in the **Line Width** spin control. The default is 1 point.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Fixed Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Button Control Properties

A push button (BUTTON) is a rectangular area containing text and/or a picture. When the user presses the button, it should execute a command described by the text or picture.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** property or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

Text color only affects button text when an icon is present. However, Text color always affects the focus indicator (dotted rectangle that appears when the button has focus).

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Suppress Transparency 	Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.
UseVisualStyles	Set to TRUE to display this control in a Visual Style appearance (as if the XP manifest was active)

Extra

Delay	The amount of time before the second EVENT:Accepted occurs in hundredths of seconds. The first EVENT:Accepted occurs when the end user clicks and holds the button. (The DELAY

Drop ID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Icon In the **Icon** field, optionally select a standard icon, .BMP, or .GIF file. This displays a small bitmap on the button face (clipping or centering the bitmap as necessary).

To select a standard icon, choose one of the named items in the drop-down list. To select an icon file (whose extension must be .ICO), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Repeat This is how often the EVENT:Accepted occurs in hundredths of seconds when the end user clicks and holds the button. (The REPEAT attribute)

STD ID Executes a standard action when the end user presses the button. **See Also:** STD .

General

Justification The **Justification** drop down list is used with button icons. ***Default*** places the icon above any text. ***Right-Justified*** places the icon to the right of any text. ***Left-Justified*** places the icon to the left of any text.

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text The text to display on the button. Place an ampersand (&) before the character to act as the accelerator key for the button--this underlines the character as it appears on the button.

Microsoft recommends you do *not* place an accelerator key on buttons labeled 'OK,' or 'Cancel.'

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. The field equate label references the button in program statements.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the mouse over the button. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the button has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

Default Button "Presses" the button when the user presses the ENTER key. A heavy border appears around the button at runtime, to signal the default button to the user. In general, place the DEFAULT attribute on a button if it represents the most likely action the user will wish to carry out. Place only one default button in a window.

Flat (the FLAT attribute) creates a button control which appears flat until the mouse cursor passes over it. This is typically used on toolbar buttons. This feature works best if the ICON attribute names a .GIF file.

Immediate Lets you create a button control which repeats the executable action continuously, for as long as the user holds the button down. Normally, buttons generate an event only after the user presses *and releases* the mouse. **See Also:** the IMM attribute.

Required Specifies that when pressed, your program automatically checks that all entry controls with the REQ attribute are neither blank nor zero. A button with this attribute is a 'required fields check' button.

Specify this type of button when a window also contains an ENTRY or TEXT control field with the REQ attribute (or else use the INCOMPLETE() function to test the ENTRY controls). When the user presses a button with the REQ attribute and an ENTRY field is blank or zero, the first required control which is blank or zero receives the focus.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Actions

Use the Actions tab to provide functionality to your button. Filling in these prompts causes the button to execute an action when the user presses the button.

When Pressed	From the drop down list, choose Call a Procedure, Run a Program, or No Special Action.
	Choose No Special Action if the button is from a control template.
	The procedure or program you specify executes when the user pushes the button. The choices are:
Call a Procedure	You must specify the **Procedure Name**, and whether the procedure will **Initiate a Thread**.
Procedure Name	From the **Procedure Name** drop down list, choose an existing procedure name, or type a new procedure name. A new procedure appears as a "ToDo" item in your Application Tree.
Initiate a Thread	Optionally check the **Initiate a Thread** box. If the procedure initiates a thread, specify the Thread Stack size. Clarion uses the START function to initiate a new execution thread. You can additionally specify **Parameters**, **Requested File Action**, or both.
	A BUTTON on an application frame toolbar that calls an MDI child procedure must initiate a thread.
Thread Stack	Accept the default value in the **Thread Stack** spin box unless you have extraordinary program requirements. To change the value, type in a new value or click on the spin box arrows.
Parameters	In the **Parameters** field, optionally type a list of variables or data structures passed to the procedure. You are limited to the number of passed parameters supported by the START statement. Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the appropriate prompt.
Requested File Action	From the drop down list, optionally select ***None, Insert, Change, Delete***, or ***Select***. The default selection is ***None***. The Global Request variable gets the selected value. The called procedure can then check

	the value of the Global Request variable and perform the requested file action.
Run a Program	You must specify the **Program Name**, and optionally, any parameters.
Program Name	Type the program name. The program name must be in your path or current folder, else enter the full path and executable program. Quotes are added to your entry so you don't need to enter any.
Parameters	Optionally type a list of values that are passed to the program. Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the appropriate prompt. The variable must be preceded by an exclamation point (!).
No Special Action	Choose this option if you are providing your button's functionality with another method, such as embedded source, or an STD ID (see *Extra Tab* above).
	You may combine a procedure or program call with embedded source, but not with an STD ID.
Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Check Box Control Properties

The check box provides an attractive way to display a yes/no choice for a record field--the alternative might be an entire column that repeats "one," "yes," or even ".T." for each record.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **Text Color** or **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Drop ID — To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Icon — In the **Icon** field, optionally select a standard icon or icon file. This displays a small bitmap next to the check box (clipping or centering the bitmap as necessary).

To select a standard icon, choose one of the named items in the drop-down list. To select an icon file (whose extension must be .ICO), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

General

Justification — **Left Justification** arranges the check box (or icon) to the left of the parameter text. **Right Justification** arranges the check box (or icon) to the right of the parameter text. **Default** arranges the check box according to any applicable settings in the data dictionary.

Layout — Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left.

The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text — Specify a string constant to display by typing it in the **Text** box.

TextFont — Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use — This defines the USE attribute for the control. Specify a numeric variable. The check box places a value of 1 in the numeric variable if the end user turns on the check box, zero if off.

Value — These values are also used to set the state of the check box (checked or cleared) when it is first displayed.

True Value — Type the value to assign when the box is checked.

False Value — Type the value to assign when the box is cleared.

True Value and **False Value** let you easily manage legacy data with a check box, or let you use character values such as "T" and "F" or "Yes" and "No" where appropriate. For example, if your legacy field contains "True" and "False" or "Y" and "N," rather than 1 and 0, then **True Value** and **False Value** can modify the check box's default behavior to be consistent with the legacy data. If you leave both fields blank, you get the default values and behavior, that is, 1 for checked and 0 or blank for cleared.

True Value and False Value are case sensitive, so "True" is not the same as "TRUE" and "T" is not the same as "t." Also, if you are using a STRING variable as the USE variable for your check box, you must explicitly set True and False Values (here, or in the dictionary's validity check for the target column).

Mode

Disable — Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use

the ENABLE statement to allow the user access to the control

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control. .

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control.

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip	The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Options

Flat	The FLAT attribute (PROP:FLAT) specifies the BUTTON, CHECK, or RADIO with an ICON attribute appears flat until the mouse cursor passes over it. With check boxes, this creates a "latched" button effect.

Position

Lets you set the location and size of the control.

The **Position** category lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Actions

The Check Box **Actions** category leads to other dialogs allowing you to name variables and change their values when the end user checks or unchecks the box. Additionally, you can HIDE or UNHIDE other controls in the window.

Two group boxes with two pairs of buttons appear on the **Actions** tab. These buttons set the behavior for **When the Check Box is Checked**, and **When the Check Box is Unchecked**.

Assign Values	Opens the **Assign Values** dialog where you can assign values to variables based on the checked or unchecked state of the check box.
Hide/Unhide Controls	Opens the **Hide/Unhide Controls** dialog where you can specify window controls to hide or unhide based on the checked or unchecked state of the check box.
Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Assign Values Dialog

Lets you assign values to variables based on the checked or unchecked state of a check box. You may specify multiple assignments. Press the **Insert** button to add a new assignment.

Variable to Assign In the entry box, type a variable name, or press the ellipsis (...) button to choose or create a data dictionary field or a memory variable with the **Select Field** dialog.

Value to Assign In the entry box, type the value to assign to the variable. You can then add code to your program to take appropriate action based on the run time value of the variable(s). Press the **E** button to call the Expression Editor. This dialog is used to help you construct syntactically correct expressions to use in the appropriate prompt.

Hide/Unhide Controls Dialog

Lets you specify window controls to hide or unhide based on the checked or unchecked state of a check box. You may specify multiple controls to hide/unhide. Press the **Insert** button to add a new hide/unhide action to the list.

Control to hide/unhide From the drop down list, choose the control to HIDE or UNHIDE.

Hide or unhide control From the drop down list, choose **Hide** or **Unhide**.

Combo Box Control Properties

The Combo Box combines an entry box with a list box. It is useful for when you expect string data which *usually* should be a member of the list, but which also might not be. The **Window Designer** lets you create either a normal combo box, or a drop-down combo box.

This section only discusses *placing* the combo box. After you place it, you must format it. See the **List Box Formatter** dialog for more information on formatting and adding additional functionality to your combo boxes.

See Also:

How to Use a Combo Box

FileDropCombo Control template

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view

Color

Enter a valid color equate in the TextColor , BackGround, Selected text, Selected fill, or GridColor properties, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.
UseVisualStyles	Set to TRUE to display this control in a Visual Style appearance (as if the XP manifest was active)

Extra

Case	Specify case attributes for the entry field portion of the combo box. The entry box can automatically convert characters from one case to another. **Uppercase** automatically converts to all caps. **Capitalize** converts to proper case. **Default** (no attribute) accepts input in the case the user

types it.

Column Set to TRUE if you wish to allow the user to highlight the list component of a multi-column combo box field by field (rather than one row at a time). This provides for spreadsheet grid style movement of the highlight bar. **See Also:** the COLUMN attribute.

DragID To specify the type of Drag operations this control will generate, type up to 16 *signatures*, separated by commas. The **DRAGID** attribute specifies the REGION control can serve as a drag-and-drop host. DRAGID works in conjunction with the **DROPID** attribute.

DropID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

EntryMode Choose either **Insert**, **Overwrite** or **Default**. The **Entry Mode** applies only for windows with the MASK attribute set. Default accepts input according to the current system settings.

Imm To generate a message event each time the end user moves or resizes the selection bar, set this property to TRUE. This adds the IMM attribute to the COMBO. You are responsible for the code that executes upon notification of the event.

Mark Optionally enables multiple item selection. Type in the name of a QUEUE, field or array in the **Mark** field if you wish to allow the user to select more than one item from the list. The QUEUE field will flag the selected items.

Read Only The READONLY attribute prevents data entry in this control. Use this to declare display-only data.

Required The REQ attribute specifies that the control may not be left blank or zero.

VCR Optionally provide VCR controls. Set the **VCR** property to TRUE to provide VCR style controls for the list box. Optionally type the name of an entry **Field**. When the user presses the *Locator* (?) button, the focus shifts to that field. The user may type in data, then press TAB to scroll the list box to the closest matching entry.

General

Drop Specifies whether this should be a regular or drop-down combo box. Place a zero in the **Drop** field for a normal combo box. To create a drop-down combo box, type the number of drop-down elements you wish to be visible. You must resize the combo box after specifying the drop number.

Format Specifies the print or dispay format of the list box control. Press the ellipsis to call the *Format Editor* dialog.

From Fill the **From** field with the origin of the list data. Generally, this is the label of a QUEUE structure.

Justification Specify left, center, right, decimal, or default justification. Default justification matches that specified in the data dictionary, if applicable. If you use decimal justification, you set the Offset to allow display of digits to the right of the decimal point.

Offset Specify an indentation value for the list box item text, in dialog units.

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

NoBar Specifies the selection bar appears only when the list box has focus. **See Also:** the NOBAR attribute.

Picture Specify the picture token for the control. The picture token you specify appears in the format string, for example, "@S10@." Pressing the ellipsis button lets you select the picture token from the **Edit Picture String** Dialog.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Place a variable or field equate label in the **Use** field. You may specify the variable which will receive the value that the user selects. Or, a field equate label to reference the combo box in program statements.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control.

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control. .

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

Flat	Check this box to give your list box a "flat" appearance (the list box control is on the same level as the window, and not recessed). Due to a built-in attribute to the runtime library, list boxes used in Drop List and Drop Combo controls are always flat and cannot be modified in the Formatter.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

ScrollBars

Horizontal	To add a horizontal scroll bar to your list box, set the **Horizontal** property to TRUE. Scroll bars only appear when the list of items in the list is bigger than the window.
Vertical	To add a vertical scroll bar, set the **Vertical** property to TRUE.

These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the control.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Ellipse Control Properties

The ellipse control lets you place a circle or ellipse in your window or report. You may fill the ellipse with a color, and specify a border color. The ellipse control cannot receive focus, nor can it generate events.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **BorderColor** or **FillColor** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

General

Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
Line Width	Set the thickness of the ellipse's border by entering a point value in the **Line Width** spin control. The default is 1 point.
Use	This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Mode

Disable	Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.
Hide	Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.
Scroll	Specifies whether the control should move with the window when the user scrolls the window. By default (False), the control does not move with the window. Leave the **Scroll** property False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the control.

The **Position** category lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Entry Box Control Properties

An entry box lets you process data input from the user. The data entry control is a specialized form of Windows edit box. It can help you automatically validate data as the user enters it in a dialog box.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the TextColor, BackGround, SelForeGround, or SelBackground fill fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

DropID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.
EntryMode	Choose either **Insert**, **Overwrite** or **Default**. The **Entry Mode** applies only for windows with the MASK attribute set. Default accepts input according to the current system settings.
TextCase	Specify case attributes for the entry field. The entry box can automatically convert characters from one case to another. **Uppercase** automatically converts to all caps. **Capitalize** converts to proper case. **Default** (no attribute) accepts input in the case the user types it.

General

Justification Specify left, center, right, decimal, or default justification. Default justification matches that specified in the data dictionary, if applicable. If you use decimal justification, you set the Offset to allow display of digits to the right of the decimal point.

Offset Specify an indentation value for the text, in dialog units. The indention is in the opposite direction from the justification.

Layout Indicates the orientation of the control.

Left to Right maintains the original layout specified in the Window Designer.

Right to Left essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left.

The setting in the Application Frame will cascade its setting to all child windows and controls that have the *default* setting active.

Picture

The Picture field takes a display picture token that specifies input format. You may press the ellipsis (...) button next to the field to pick a display picture from the **Edit Picture String** dialog.

You may check the user entry against the picture at two points: as the user types the data in, or when the user closes the dialog box. Checking the data as the user types it incurs a slight performance penalty. To do so, check the **Entry Patterns** box in the **Window Properties** dialog for the window in which the entry box resides. This turns the MASK attribute on for *all* controls in the window.

If the MASK attribute is off, entry checking takes place when the user moves the focus to another control (for example, by TABBING to another field).

If the user types in data in a format different from the picture, the program will attempt to determine the format, then convert it to match the picture (if no MASK was specified). For example, if the user types 'January 1, 1995' and the picture is @D1, the program formats the input to "1/1/95. If the program cannot determine the entry format, it will *not* update the USE variable. The user will receive an audible prompt (beep), and the focus will return to the entry control, ready for additional input.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Place a variable or field equate label in the **Use** field. You may specify a variable which receives the value that the user types. Or, a field equate label which references the entry box in program statements.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent	Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

Set the Entry properties. There are five properties you may set independently.

Flat	Specifies a "flat" appearance (the entry control is on the same level as the window, and not recessed).
Immediate	The IMM attribute specifies immediate event generation whenever the user presses any key. **See Also:** How to complete an entry field when the last character is entered.
Password	The PASSWORD attribute specifies non-display of data entered in this control. When the user types in data, asterisks are displayed for each character entered.
Read Only	The READONLY attribute prevents data entry in this control. Use this to declare display-only data.
Required	The REQ attribute specifies that the control may not be left blank or zero.

Position

Lets you set the location and size of the control.

The **Position** category lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Actions

The **Actions** tab prompts are all from the templates, in other words, the prompts you see here vary with the template used to create the control. Following are the standard action prompts for all entry controls.

The standard **Actions** prompts are designed to provide data validation support for your entry controls. The tab is divided into two parallel sections. The **When the Control is Selected** section provides validation when the control *receives* focus (when the user TABS onto, or mouse CLICKS the control). The **When the Control is Accepted** section provides data validation when the control *loses* focus after data have been entered in it. The control loses focus when the user TABS off

the control, mouse CLICKS to a different control or window, or closes the window without canceling. The two sections are not mutually exclusive, so you can provide validation at both points.

Lookup Key

Type a key label from the *lookup* file, or press the ellipsis (...) button to select a key from the **Select Key** dialog.

A lookup file is a file which contains all the valid values for the entry field, which are directly accessible through a unique key.

For example, a file containing all of the customer numbers for your application could be a lookup file. The key label could be "CUS:KeyCustNumber".

This lookup validation is easier to implement by using a single component unique key.

Lookup Field

Type the label of a component field of the lookup key, or press the ellipsis (...) button to select a field from the **Select component from key** dialog.

This is the field within the key that contains the same value being validated. Ideally, this field is the only component of a unique key.

Lookup Procedure

Type a procedure name, or choose an existing procedure from the drop down list.

This is the procedure that is called when the user enters an invalid value, and the lookup specified above *fails*. The usual purpose of this procedure is to allow the user to choose a valid value from the lookup file.

Select procedures (or Browse procedures) generated by Clarion's Wizards) are appropriate for this purpose. Alternatively, you may hand-code a procedure.

Procedure Parameters

Allows you to specify parameter names (an optional list of variables separated by commas) for your update procedure, which you can pass to it from the calling browse procedure. You must specify the functionality for the parameters in embedded source code.

Example: (LOC:HideID,GLO:AccessLevel)

Advanced

Calls the **Embedded Source** dialog. The only embed point shown is after the code generated to call the lookup procedure specified above. For more embed points, and further customization, press the **Embeds** button.

Perform lookup during non-stop select

Checking this box tells Clarion to perform the validation when the *window* is accepted, even if the *entry control* never received focus. From a practical viewpoint, checking this box prevents the user from entering blanks by virtue of having pressed the window's "**OK** button" without ever TABBING or CLICKING onto the entry field.

This option is only applicable to the **When the Control is Accepted** section.

Force Window Refresh when Accepted

Checking this box ensures that everything (including formulas and other entry fields) on the window is current and up-to-date when the user TABS off this entry control.

More Field Assignments

Optionally press the **More Field Assignments** button to specify additional value assignments from the selected lookup item's record.

Embeds

Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Group Control Properties

A Group control places a box around two or more controls. It visually associates the controls for the user, and lets you address all the controls as one entity -- making it easy, for example, to disable all at once.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Suppress Transparency 	Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Bevel	Gives a three dimensional look to the control. The box appears raised, depressed, or both. The **Window Designer** adds the BEVEL attribute to your control declaration. Fine tune the bevel with the following prompts:
Outer	A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins at the outer border of the box.
Inner	A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the

window. The bevel effect begins immediately inside the outer bevel.

Style An integer constant or constant expression that specifies fine control of the bevel, overriding the signs of the outer and inner parameters (PROP:BevelStyle).

Boxed Specifies whether to draw a visible box, containing the caption, around the grouped controls. When False, the Group box, including its caption, is invisible.

Drop ID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text Specify a string constant by typing it in the **Text** box.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default, (False), the control does not move with the window. Leave the **Scroll** box False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Position

Lets you set the location and size of the control.

The **Position** category lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Image Control Properties

The Image control lets you place bitmapped and vector images in a window or report. The bitmap file formats supported are .BMP, .PCX, .GIF, .ICO, .CUR and .JPG. The vector file format supported is .WMF. Clarion can support up to 16.7 million color resolution.

You must add the PALETTE attribute to a WINDOW to support custom palettes and color depths beyond the resolution of the end user's machine at runtime. For example, to support a 16.7 million color palette, the proper attribute is PALETTE(16777215).

The Image control cannot receive focus, nor can it generate events.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

General

File — Select a graphics file. Type in a file name, or press the ellipsis (**...**) button to the right of the **File** field to select a graphics file using the standard **File Open** dialog.

Image Mode — Select from the drop list one of the following modes:

Stretched — Specifies the image displays stretched to fill the area of the IMAGE.

Centered — Specifies the image displays at its default size and is centered in the area of the IMAGE (PROP:CENTERED)

Tiled — Specifies the image displays at its default size and is tiled to fill the entire area of the IMAGE (PROP:TILED).

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default, (False), the control does not move with the window. Leave the **Scroll** property False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Fixed Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Scroll Bars

Horizontal	To add a horizontal scroll bar to the control, set the **Horizontal** property to True. Scroll bars only appear when the contents of the text box are bigger than the window.
Vertical	To add a vertical scroll bar, set the **Vertical** property to True.

These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the control.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Line Control Properties

The line control lets you place a straight line in your window or report. You specify a color. The line control cannot receive focus, nor can it generate events.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Colors

Enter a valid color equate in the **LineColor** property, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

General

Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
Line Width	Set the line's thickness by entering a point value in the **Line Width** spin control. The default is 1 point.
Use	This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Mode

Disable	Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.
Hide	Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.
Scroll	Specifies whether the control should move with the window when the user scrolls the window. By default, (False), the control does not move with the window. Leave the **Scroll** property False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

A height of zero(0) specifies a horizontal line. A width of zero(0) specifies a vertical line.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

List Control Properties

The List box is useful for presenting a great number of choices for the user. It can convey a large amount of data in a small area, which has led to its use as an all purpose data control. Using Clarion, you can create list boxes which look like spreadsheet grids, perform drag-and-drop tasks, and more.

This section only discusses *placing* the list box. After you place a list box, you must format it. See the **List Box Formatter** dialog for more information on formatting and adding additional functionality to your list boxes.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor**, **Background**, **SelForeGround**, **SelBackground**, or **GridColor** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Design settings control visual elements you see in the Window Designer. These settings should not affect the window display at runtime.

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.
UseVisualStyles	Set to TRUE to display this control in a Visual Style appearance (as if the XP manifest was active)

Extra

Column	Set this property to TRUE if you wish to allow the user to highlight a multi-column list box field by field (rather than one row at a time). This provides for spreadsheet grid style movement of the highlight bar. **See Also:** the COLUMN attribute.

DragID To specify the type of Drag operations this control will generate, type up to 16 *signatures*, separated by commas. The **DRAGID** attribute specifies the LIST control can serve as a drag-and-drop host. DRAGID works in conjunction with the **DROPID** attribute.

DropID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Imm To generate a message event each time the end user moves or resizes the selection bar, set **Imm** to TRUE. This adds the IMM attribute to the LIST. You are responsible for the code that executes upon notification of the event.

Mark Optionally enables multiple item selection. Type in the label of a QUEUE field in the **Mark** field if you wish to allow the user to select more than one item from the list. The QUEUE field will flag the selected items.

VCR Optionally provide VCR controls. Set **VCR** property to TRUE to provide VCR style controls for the list box. Optionally type the field equate label of an entry field to the right of the check box. When the user presses the *Locator* (?) button, the focus shifts to that field. The user may type in data, then press TAB to scroll the list box to the closest matching entry.

General

Drop Specifies whether this should be a regular or drop-down list box.

Count Place a zero in the **Count** property for a normal list box. To create a drop-down list box, type the number of drop-down elements you wish to be visible.

Width When the Drop field is non-zero, specify an integer constant here that sets the width of the dropped list, in dialog units.

Format Specifies the print or display format of the data in the list control. Press the ellipsis button to call the **Format Editor**.

From Fill the **From** field with the origin of the list data. Generally, this is the label of a QUEUE structure.

Justification Specify left, center, right, decimal, or default justification. Default justification matches that specified in the data dictionary, if applicable. If you use decimal justification, you set the Offset to allow display of digits to the right of the decimal point.

Offset Specify an indentation value for the list box item text, in dialog units.

Layout Indicates the orientation of the LIST control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

NoBar Specifies the selection bar appears only when the list box has focus. **See Also:** the NOBAR attribute.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Place a variable or field equate label in the **Use** field. You may specify the variable which will receive the value that the user selects. Or, a field equate label to reference the list box in program statements.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these

keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message — The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip — The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable — Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide — Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll — Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip — Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent — Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options:

Flat — Set this property to True to give your list box a "flat" appearance (the list box control is on the same level as the window, and not recessed). Due to a built-in attribute to the runtime library, list boxes used in Drop List and Drop Combo controls are always flat and cannot be modified in the Formatter.

Position

Lets you set the location and size of the control.

The **Position** property lets you specify the AT attribute. Filling in the attribute manually is optional--you may also set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Fixed	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Scrollbars

Horizontal	To add a horizontal scroll bar to your list box, set the **Horizontal** property to TRUE.
Vertical	To add a vertical scroll bar, set the **Vertical** property to TRUE.

Scroll bars only appear when the list of items in the list is bigger than the window. These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the control.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

OLE Properties

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the Text Color or Background fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra:

DropID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Compatibility	Set to True to specify a compatibility mode of 1 for objects that require it (such as the Windows bitmap editor). Set to False to let the compatibility mode default to 0.
ControlType	Select the object type for the OLE control declaration. Select from **Ole** or **Document**

When *OLE* is the Control Type:

ServerName The **ServerName** list contains registered OLE objects. The ControlType property gets the CREATE attribute if no **Storage File** is specified or them OPEN attribute if a **Storage File** is specified. Select from a list of registered OLE objects, such as Excel Spreadsheets, Word documents, PowerPoint Slides, etc. to CREATE or OPEN.

Note:

When you select an OLE server, the server is automatically loaded by the Window Designer. This can be time consuming during the window design process. Design and draw your window first, then specify your OLE control last, or specify the server at run-time using property syntax rather than at design time.

StorageFile Specifies the name of an OLE Compound Storage File and the object within it to OPEN. Separate the filename and the object name with a backslash and exclamation point: *FileName\!ObjectName.* The Application Generator supplies a default value for this field when you use the development environment to create the Compound Storage File.

Note:

RIGHT-CLICK the OLE control in the Window Designer and choose Open from the popup menu to activate the specified OLE server in open mode. This lets you build the object independent of your application, and it automatically specifies a default filename\!objectname in the Storage File field. When a Storage File is specified, the object is OPENed rather than CREATEd.

The OLE Server can access and manipulate the object in the storage file, but only through your application. The OLE Server cannot access the storage file independently of your application. Use a Storage File when you want to limit your user's access to the document.

When the object is opened, the saved version of the OLE container properties are reloaded, so properties need not be specified for OPENed objects.

If this field is blank, the OLE object is CREATEd rather than OPENed.

When *Document* is the Control Type:

Document The **Document** property replaces the **ServerName** list. The ControlType property gets the DOCUMENT or LINK attribute. Type the fully qualified name of the document file or press the ellipsis button (...) to select the file from the standard dialog. A document file is a file that is associated with a specific OLE server, so the application can activate that server at runtime (e.g.. MYBUDGET.XLS is associated with Excel). If the filename is not fully qualified, the application looks for it in the current directory. The document file should be installed on the end user's machine in the specified directory.

StorageFile Specify the name of an OLE Compound Storage File and the object within it to OPEN. Separate the filename and the object name with a backslash and exclamation point: *FileName\!ObjectName.* A Storage File may be specified for embedded documents (DOCUMENT) but not for linked documents (LINK). The Application Generator supplies a default value for this field when you use the development environment to create the Compound Storage File.

Note:

RIGHT-CLICK the OLE control in the Window Designer and choose Open from the popup menu to activate the specified OLE server in open mode. This lets you build the object independent of your application, and it automatically specifies a default filename\!object name in the Storage File field. When a Storage File is specified, the object is OPENed rather than CREATEd.

The OLE Server can access and manipulate the object in the storage file, but only through your application. The OLE Server cannot access the storage file independently of your application. Use a Storage File when you want to limit your user's access to the document.

When the object is opened, the saved version of the OLE container properties are reloaded, so properties need not be specified for OPENed objects.

If this field is blank, the OLE object is CREATEd rather than OPENed.

Link Set this property to TRUE to make the OLE object a link. This generates the LINK attribute for the OLE control, which tells the server to update the original file with any changes made through your application. Set this property to False to make the OLE object an embed. This generates the DOCUMENT attribute for the OLE control, which tells the server *not* to update the original file with changes made through your application. The default DoVerb action ({PROP:DoVerb}=0) may depend on whether the object is a link or an embed.

General (continued):

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

SizingMode Select a sizing attribute from the drop-down list. The **Window Designer** adds the attribute to the OLE declaration.

Default Adds no sizing attribute. Zoom is the default.

Clip The OLE declaration gets the CLIP attribute. The OLE object only displays what fits into the area defined by the OLE container control's AT attribute. If the object is larger than the control, only the top left corner displays.

Stretch The OLE declaration gets the STRETCH attribute. The OLE object stretches to completely fill the area defined by the OLE container control's AT attribute. The object's aspect ratio is lost.

AutoSize The OLE declaration gets the AUTOSIZE attribute. The OLE object automatically resizes when the OLE container control's AT attribute changes at runtime.

Zoom The OLE declaration gets the ZOOM attribute. The OLE object stretches to fill the area defined by the OLE container control's AT attribute. The object's aspect ratio is maintained.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Help

Cursor The **Cursor** field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Enter the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the text should not be longer than

can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Value Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event

handling for this control only.

Option Control Properties

The **OPTION** control declares a group of RADIO controls that offer the user a list of mutually exclusive choices. The multiple RADIO controls in the OPTION structure define the choices offered to the user.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Suppress Transparency  — Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Bevel — Gives a three dimensional look to the control. The control appears raised, depressed, or both. The **Window Designer** adds the BEVEL attribute to your control declaration. Fine tune the bevel with the following prompts:

Outer — A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins at the outer border of the box.

Inner A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins immediately inside the outer bevel.

Style An integer constant or constant expression that specifies fine control of the bevel, overriding the signs of the outer and inner parameters (PROP:BevelStyle).

Boxed The **BOXED** attribute specifies a single-track border around the OPTION structure. The parameter text of the OPTION control appears in a gap at the top of the border box. If BOXED is omitted, the *text* parameter is not displayed on screen

Drop ID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text Specify a string constant by typing it in the **Text** box. If the control is to display a variable, type a picture token in this box.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using

the standard file dialog.

HelpID — The **HelpID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key — Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message — The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip — The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable — Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide — Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll — Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip — Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent — Specify whether you wish the control background to be **Transparent**. This

instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Position

Lets you set the location and size of the control.

The **Position** properties let you specify the AT attribute. Filling in the attribute manually is optional--you may also set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Value	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Panel Properties

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **FillColor** property, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds FILL attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Suppress Transparency  — Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Bevel — Gives a three-dimensional look to the control. The control appears raised, depressed, or both. The **Window Designer** adds the BEVEL attribute to your control declaration. Fine tune the bevel with the following prompts:

Outer — A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins at the outer border of the box.

Inner — A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins immediately inside the outer bevel.

Style	An integer constant or constant expression that specifies fine control of the bevel, overriding the signs of the outer and inner parameters (PROP:BevelStyle).

General

Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
Use	This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Mode

Disable	Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.
Hide	Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.
Scroll	Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the control.

The **Position** property lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Value	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Progress Control Properties

The **PROGRESS** control declares a control that displays a progress bar. This usually displays the current percentage of completion of a batch process.

If a variable is named as the USE attribute, the progress bar is automatically updated whenever the value in that variable changes. If the USE attribute is a field equate label, you must directly update the display by assigning a value (within the range defined by the RANGE attribute) to the control's runtime property (PROP:Progress).

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Design settings control visual elements you see in the Window Designer. These settings should not affect the window display at runtime.

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.
UseVisualStyles	Set to TRUE to display this control in a Visual Style appearance (as if the XP manifest was active)

Extra

Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.
Range	Specifies the range of values the progress bar displays. If omitted, the default range is zero (0) to one hundred (100).

General

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. The field equate label references the control in program statements.

Help

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

Vertical	Set the **Vertical** property to True to allow the progress control to operate from the bottom of the control to the top. If your progress control is positioned in a horizontal (left to right) display format, you should resize the progress control accordingly.
Smooth	Set the **Smooth** property to True to allow a smooth incremental display of the progress control instead of the standard "block" format.

Position

Lets you set the location and size of the control.

The **Position** property lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Value	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Prompt Control Properties

The Prompt control lets you place a specialized string object on screen which automatically provides an accelerator or mnemonic access key to the next active control following the prompt.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Suppress Transparency 	Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Justification	Specify left, center, right, or default justification. Default justification matches that specified in the data dictionary, if applicable.

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text Specify a string constant by typing it in the **Text** box. If the control is to display a variable, type a picture token in this box.

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable

Help

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the mouse over the PROMPT. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Value	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Radio Button Control Properties

A Radio button, also called an option button, provides the user one of a set of mutually exclusive choices. By default, a filled-in circle represents the current selection.

A group box--an OPTION structure--must always surround the radio button choices. The **Window Designer** automatically prompts you to create this if you try to place a radio button outside an option box. When the user selects a choice, the control fills the USE variable with the Radio text, minus the ampersand indicating the accelerator key.

When you place a radio button in a blank dialog window, it forces the creation of the OPTION STRUCTURE. After activating the Radio Button tool, or choosing **Radio Button** from the **Control** menu, CLICK in the window. An option box and one radio button appear. Select the radio button and press the Properties button, or RIGHT-CLICK and select **Properties** to open the **Radio Button Properties** dialog.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Icon

In the **Icon** field (ICON), optionally select a standard icon or icon file. This displays a small bitmap on the button face (clipping or centering the bitmap as necessary).

To select a standard icon, choose one of the named items in the drop-down list. To select an icon file (whose extension must be .ICO), choose **Select File** from the drop-down list, then pick the file using the standard file dialog. At run time, the radio button appears as a "latched" 3D push button.

General

Justification

Left Justification arranges the button (or icon) to the left of the parameter text. **Right Justification** arranges the button (or icon) to the right of the parameter text. **Default** arranges the button according to any applicable settings in the data dictionary.

Layout

Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Text

Specify a string constant to display. Place an ampersand (&) before the single character to set the accelerator key or mnemonic access character for the radio button--this underlines the label that appears next to the radio button.

TextFont

Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use

This defines the USE attribute for the control. The field equate label references the radio button in program statements.

Value

When the user selects a radio button, the OPTION's USE variable receives the value that you specify here. The value you enter should match the data type of the OPTION's USE variable.

If you leave the **Value** field blank, the OPTION's USE variable receives either the string found in the **Text** field, or the button number, depending on the data type of the OPTION's USE variable.

The button number corresponds to the button's position within the option box. From the **Window Designer** choose **Edit ▸Order Control dialog** to see the button's tab order position within the option box.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

HelpID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

Flat The FLAT attribute (PROP:FLAT) specifies the BUTTON, CHECK, or RADIO with an ICON attribute appears flat until the mouse cursor passes over it. With radio buttons, this creates a "latched" button effect.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Value Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Region Control Properties

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **BorderColor** or **FillColor** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Suppress Transparency  — Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Bevel — Gives a three dimensional look to the control. The control appears raised, depressed, or both. The **Window Designer** adds the BEVEL attribute to your control declaration. Fine tune the bevel with the following prompts:

Outer — A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins at the outer border of the box.

Inner A positive value makes the control appear raised above the plane of the window. The higher the value, the further the box is raised. A negative value makes the control appear depressed below the plane of the window. The bevel effect begins immediately inside the outer bevel.

Style An integer constant or constant expression that specifies fine control of the bevel, overriding the signs of the outer and inner parameters (PROP:BevelStyle).

Drag ID To specify the type of Drag operations this control will generate, type up to 16 *signatures*, separated by commas. The **DRAGID** attribute specifies the REGION control can serve as a drag-and-drop host. DRAGID works in conjunction with the **DROPID** attribute.

Drop ID To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code.

Help

Cursor The CURSOR attribute lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Options:

Immediate To generate a message event each time the mouse moves over the region, check the **Immediate** box. This adds the IMM attribute to the control. You are responsible for the code that executes upon notification of the event.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Value Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Sheet Control Properties

The SHEET control declares a group of TAB controls that offer the user multiple "pages" of controls for the window. The multiple TAB controls in the SHEET structure define the "pages" displayed to the user.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **BackGround** and **TextColor** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates.

Design

Design settings control visual elements you see in the Window Designer. These settings should not affect the window display at runtime.

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See Application Options.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.
UseVisualStyles	Set to TRUE to display this control in a Visual Style appearance (as if the XP manifest was active)

Extra

Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the DROPID attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Scrolling Use the drop-down list to specify tabs scrolling behavior by choosing one of the following selections:

NoScrolling The tabs do not scroll. This is the default.

JoinedButtons Tabs are scrollable, with adjacent scroll buttons. adds the JOINED attribute to the SHEET. See the *Language Reference* for more information.

SpreadButtons Tabs are scrollable, with scroll buttons at opposite ends of the sheet. Adds the HSCROLL attribute to the SHEET. See the *Language Reference* for more information.

General

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Tab Location Use the drop-down list to set the location of the tabs for this SHEET. Choose from the following:

TabPosition

Default The tabs run across the top of the sheet.

Left The tabs run down the left side of the sheet. This adds the LEFT(width) attribute to the SHEET.

Right The tabs run down the right side of the sheet. This adds the RIGHT(width) attribute to the SHEET.

Above The tabs run across the top of the sheet. This adds the ABOVE(width) attribute to the SHEET.

Below The tabs run across the bottom of the sheet. Adds the BELOW(width) attribute to the SHEET.

Width Specify the tab width in dialog units.

This sets the value of the width parameter for the LEFT, RIGHT, ABOVE, or BELOW attribute. By setting this width, you can make all your tabs the same size, regardless of varying text lengths per tab.

Tab width is the distance between the edges of the tab that are perpendicular to the text orientation. That is, width determines how much space appears on either end of your tab's text, not how much space appears above and below the text. This is true, regardless of text orientation.

Tabs Press the ellipsis button to call the Collection Editor. Here, you can modify the properties of each individual tab

TextFont Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Text Orientation Use the drop-down list to set the orientation of the tabs and their text. A **Text Orientation** other than *Default* requires a TrueType font. Choose from the following orientations:

Default The text reads left to right and the shape of the tab is a horizontal rectangle.

Up The text reads from bottom to top and the shape of the tab is a vertical rectangle. This adds the UP attribute to the SHEET.

Down The text reads from top to bottom and the shape of the tab is a vertical rectangle. This adds the DOWN attribute to the SHEET.

Inverted The text is upside down and the shape of the tab is a horizontal rectangle. This adds the UP and the DOWN attribute to the SHEET.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable.

To fit many tabs on a sheet, set the Justification to Above and set the Text Orientation to Up. Alternatively set the Justification to Right and set the Text Orientation to Default. Also, see Extra Tab to make scrolling tabs!

Help

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Key Enter the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Position

Lets you set the location and size of the sheet and the tabs on the sheet.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the window.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window.

Value Lets you set a precise width or height in Dialog units.

TabOptions

Immediate Specifies that EVENT:NewSelection generates whenever the user clicks on a TAB (even when that TAB is already the currently selected TAB). This can be useful when there are multiple SHEET controls on the same window.

NoSheet Set to True to erase the borders of the tab pages so that only the protruding selectable portion of the tab is visible. This has the additional effect of making tabs located above the sheet, fall to the bottom of the sheet, and making tabs located below the sheet rise to the top of the sheet. This adds the NOSHEET attribute to the SHEET. See the *Language Reference* for more information.

Spread Resizes the tabs on the TABs to fill all the available space on the SHEET. The resizing algorithm considers the length of the text displayed on each tab, the number of tabs, and the available space on the property sheet.

Wizard Hides the "tab" portion of the TAB controls. Hiding the tabs aids in creating a wizard. A wizard is a window with a "tabless" SHEET control containing one or more TABS. You'll need to write the code to handle the "turning of the pages". See *How to Create a Wizard*.

Do not check this box until you are finished designing the window! Alternately, you can change the sheet to a wizard at runtime after opening the window using the `?Sheet{Prop:Wizard}=True` property assignment.

Quick Links

Embeds Accesses the Embedded Source dialog for points surrounding the event handling for this control only.

Spin Box Control Properties

Spin Boxes are specialized entry boxes that only accept values in a predefined range. They also provide the user with "increase" and "decrease" buttons, which many people like because they can use the mouse to change the value. The user can also type a value directly into the control.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the TextColor, BackGround, SelForeGround, or SelBackGround, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Delay	The amount of time before the second EVENT:NewSelection occurs in hundredths of seconds. The first EVENT:NewSelection occurs when the end user clicks and holds the arrow button. (The DELAY attribute)
DropID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.
EntryMode	Choose either **Insert**, **Overwrite** or **Default**. The **Entry Mode** applies only for windows with the MASK attribute set. Default accepts input according to the current system settings.

Range Specify the upper and lower **Range** limits, and the **Step** value. Place the highest value which the control should return in the **Range Upper** field. The value should match the **Picture** field. Place the lowest acceptable value in the **Lower** field. Place the Step value--the amount by which each press of the increase or decrease buttons should change the spin box value--in the **Step** field.

Repeat This is how often the EVENT:NewSelection occurs in hundredths of seconds when the end user clicks and holds the arrow button. (The REPEAT attribute)

TextCase Specify case attributes for the entry field. The entry box can automatically convert characters from one case to another. **Uppercase** automatically converts to all caps. **Capitalize** converts to proper case. **Default** (no attribute) accepts input in the case the user types it.

General

From The FROM attribute is optional, but is useful for values that progress in an irregular increment. You may also wish to provide the user with strings formatted as Spin Box choices when the choices are a natural progression such as the days of the week or the months of the year. Specify a QUEUE in the **From** field. This and **Range** are mutually exclusive.

Justification Specify left, center, right, decimal, or default justification. Default justification matches that specified in the data dictionary, if applicable. If you use decimal justification, you set the Offset to allow display of digits to the right of the decimal point.

Offset Specify an indentation value for the list box item text, in dialog units.

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

Picture

The Picture field takes a display picture token that specifies input format. You may press the ellipsis (...) button next to the field to pick a display picture from the **Edit Picture String** dialog.

You may check the user entry against the picture at two points: as the user types the data in, or when the user closes the dialog box. Checking the data as the user types it incurs a slight performance penalty. To do so, check the **Entry Patterns** box in the **Window Properties** dialog for the window in which the entry box resides. This turns the MASK attribute on for *all* controls in the window.

If the MASK attribute is off, entry checking takes place when the user moves the focus to another control (for example, by TABBING to another field).

If the user types in data in a format different from the picture, the program will attempt to determine the format, then convert it to match the picture (if no MASK was specified). For example, if the user types 'January 1, 1995' and the picture is @D1, the program formats the input to "1/1/95. If the program cannot determine the entry format, it will *not* update the USE variable. The user will receive an audible prompt (beep), and the focus will return to the entry control, ready for additional input.

TextFont

Calls the Font dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use

This defines the USE attribute for the control. Place a variable or field equate label in the **Use** field. You may specify a variable which receives the value that the user selects. Or, a field equate label which references the spin box in program statements.

Help

Alert

Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor

The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID

The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key

Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message

The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip

The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable

Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide

Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Skip

Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Scroll

Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Transparent	Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

You may toggle the following options on or off independently.

Flat	Specifies a "flat" appearance (the spin control is on the same level as the window, and not recessed).
Immediate	Specifies immediate event generation whenever the user presses any key.
Required	Specifies that the control may not be left blank or zero.
Read Only	Prevents data entry in this control. Use this to declare display-only data.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the control based on the applicable font and picture token.
Full	The control is the full height or width of the window or report.
Value	Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

ScrollBars

Set the appearance of the Spin control buttons by checking any combination of the two **Scroll Bar** boxes.

Horizontal	Set to True to produce larger buttons arranged side by side.
Vertical	Set to True to produce smaller, vertically stacked buttons.

If both properties are set to True, the result is the same as **Horizontal**. If both properties are False, the result is similar to **Vertical**.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

String Control Properties

The String control lets you place a string constant in a window or report. It optionally lets you substitute a variable.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **BackGround** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked — "Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

Suppress Transparency  — Allows the proper display of special static parent controls when populated "on top of" multiple tab controls. This property is set to TRUE by default, and ensures a proper display regardless of Visual Styles used. No effect on runtime window.

Tab Index — Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0).

To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu).

If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Angle — Specify the angle of the text. An angle other than zero requires a TrueType font. Adds the ANGLE attribute to the STRING. With this, you can rotate the text from its normal horizontal position through a full 360 degrees.

Tip

Make the control taller than usual to accommodate the slant of the text.

DropID — To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

Extend	**Valid in REPORT only.** Enter a valid string of attributes that are assigned to a designated REPORT control for a given document type. For more information, see EXTEND.
Tallies	**Valid in REPORT only.** The total is calculated each time a highlighted structure in the **Tallies** list prints. This list is available only from the **Report Designer** and only when the **Variable String** box is checked. Use the **Tallies** list in conjunction with the **Total Type** and **Reset On** lists on the General tab to generate a custom total field. The **Report Designer** adds the TALLY attribute to the string.

General

Justification	Specify left, center, right, decimal, or default justification. Default justification matches that specified in the data dictionary, if applicable. If you use decimal justification, you set the Offset to allow display of digits to the right of the decimal point.
Offset	Specify an indentation value for the text, in dialog units for a window, or the default measurement unit for a report. This provides the *indent* parameter for the left, center, right, or decimal, justification attribute.
Layout	Indicates the orientation of the control.**Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
IsPicture	(**Variable String)** Optionally set this property to True to specify that you want to display the contents of a variable in the string control. This opens the **Select Field** dialog where you can select or define the variable. Place a picture token in the **Picture** box, such as @s24, and name the variable in the **Use** field.
Picture	For Variable Strings (see **IsPicture**), specify the picture token for the control. Press the ellipsis button to select the picture token from the **Edit Picture String** Dialog .
Text	For Constant Strings (**IsPicture=False**), specify the constant by typing it in the **Text** box.
TextFont	Calls the **Select Font** dialog which lets you change the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.
Use	This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable. Press the ellipsis button to select or define a variable.

REPORT only properties:

Reset On This drop down list is available only from the **Report Designer**, and only when the **Variable String** box is checked. To create page totals or group totals, use the **Reset On** drop down list in conjunction with your **Total Type** selection. Reset your totals at the beginning of each **Page**, or at the beginning of any BREAK group within the report.

Total Type This drop down list is available only from the **Report Designer**, and only when the **Variable String** box is checked. Choose from this drop down list to implement one of Clarion's built in report totaling functions. Choose from ***Sum, Average, Maximum, Minimum, Count,*** and ***Page No***.

To create page totals or group totals, use the **Reset On** drop down list and the **Tallies** list on the Extra tab in conjunction with your **Total Type** selection. Totals are calculated or tallied each time a structure in the **Tallies** list prints. Totals are reset each time the **Reset On** structure prints.

Using The label of a numeric variable to receive the intermediate values calculated for the SUM, AVE, MAX,MIN, CNT, or PAGENO. This allows you to create totals on other totals. The value in the variable is internally updated by the print engine, so it is only useful for use within the REPORT structure. This is only available when a **Total Type** is specified.

Help

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Mode

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Scroll Valid only on a WINDOW. Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip Valid only in a REPORT. Check this box to specify not to print the control if the content is blank, and to move all following controls in the band upward to "fill in" the blank (PROP:SKIP).

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Value Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Tab Control Properties

The TAB structure declares a group of controls that constitute one of the multiple "pages" of controls contained within a SHEET structure. The multiple TAB controls in the SHEET structure define the "pages" displayed to the user. The SHEET structure's USE attribute receives the *text* of the TAB control selected by the user.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** and **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.

General

Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
Text	Specify a string constant by typing it in the **Text** box. If the control is to

display a variable, type a picture token in this box.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable.

Mode

Disable Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Help

Help ID The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Options:

Required

Specifies that when selected, your program automatically checks that all entry controls with the REQ attribute are neither blank nor zero.

Specify this type of tab when a window also contains an ENTRY or TEXT control field with the REQ attribute (or else use the INCOMPLETE() function to test the ENTRY controls). When the user clicks on a tab with the REQ attribute and an ENTRY field is blank or zero, the first required control which is blank or zero receives the focus. **See Also:** How to Create a Multi-Page Form .

Quick Links

Embeds

Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Text Box Control Properties

The Text control provides a multi-line data entry field. This control is especially suitable for holding a long string.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.
Tab Index	Determines the index in the TAB order that this control will follow. The first tab in the order is index zero (0). To change the **TabIndex** property *immediately* while in the Designer, use the **Show tab order** interface (i.e. on the right-click menu). If you set the **TabIndex** property using the property page, the z-order is not updated until you save and exit the Window Designer.

Extra

Boxed	The **BOXED** attribute specifies a single-track border around the TEXT structure. This attribute only works if the TEXT control is transparent or the parent window does not have the GRAY attribute applied.
Drop ID	To specify the type of Drag operations this control will accept, type up to 16 *signatures*, separated by commas. The **Window Designer** adds the **DROPID** attribute to the control, which indicates the control is a valid target for the drag and drop operations identified by the signatures.
Extend	**Valid in REPORT only.** Enter a valid string of attributes that are assigned to a designated REPORT control for a given document type. For more information, see EXTEND.

General

Justification	Specify left, center, right, or default justification. Default justification matches that specified in the data dictionary, if applicable.

Layout Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.

RTF Check this box to allow the TEXT control to read and write its contents in **Rich Text Format**. A full set of supported methods is provided to allow you to add additional features and support. See RTF.

Value Mode This drop list is enabled if you have checked the **RTF** option. Select *Field* if the actual rich text format is stored in the selected field. Select *File* if the field name is storing the file name from which the rich text format is to be read from and written to.

TextCase Specify case attributes for the entry field. The entry box can automatically convert characters from one case to another. **Uppercase** automatically converts to all caps. **Default** (no attribute) accepts input in the case the user types it.

TextFont Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.

Use This defines the USE attribute for the control. Place a variable or field equate label in the **Use** field. You may specify a variable which receives the value that the user types. When using multi-line controls, be sure the variable is large enough to hold the amount of data you expect your users to enter in the control. Or, type a field equate label which references the entry box in program statements.

Help

Alert Press the ellipsis to open a dialog that lets you add the ALRT attribute to a window or control. When the attribute is set, the window generates an EVENT:AlertKey if the user presses the key(s) you specify in this dialog, while the window has the focus.

Cursor The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.

Help ID

The **Help ID** field (the HLP attribute) takes a string constant specifying the key for accessing a specific topic in the Help document. This may be either a Help keyword or a context string.

A Help keyword is a word or phrase indexed so that the user may search for it in the *Help* **Search** dialog. When you fill in the HLP attribute for a button, if the entry box has focus, when the user presses F1, the help file opens to the referenced topic. If more than one topic matches a keyword, the search dialog appears.

When referencing a context string in the **Help ID** field, you must identify it with a leading tilde (~).

Key

Press the desired key or key combination (for example, CTRL+H). The keys you pressed will appear in the **Key** field, and will be supplied as parameters to the KEY or ALRT attribute for this control. The ESC, ENTER, and TAB keys *cannot* be specified by pressing them. For these keys, press the ellipsis (...) button and type "esc," "enter," or "tab."

Message

The **Message** field (the MSG attribute) lets you specify text to display in the first zone of the status bar when the control has focus.

Tip

The **TIP** attribute on a control specifies the text to display in a "balloon help" box when the mouse cursor pauses over the control. Although there is no specific limit on the number of characters, the *string* should not be longer than can be displayed on the screen.

Mode

Disable

Disables or 'grays-out' the control when your program initially displays it. The **Window Designer** places the DISABLE attribute on the control. Use the ENABLE statement to allow the user access to the control.

Hide

Makes the control invisible at the time Windows would initially display it. Windows actually creates the control--it just doesn't display it on screen. The **Window Designer** places the HIDE attribute on the control. Use the UNHIDE statement to display the control.

Scroll

Specifies whether the control should move with the window when the user scrolls the window. By default (False) the control does not move with the window. Set the **Scroll** property to False to create a control that stays fixed when the user scrolls the window. The **Window Designer** places the SCROLL attribute on the control when checked.

Skip

Instructs the **Window Designer** to omit the control from the Tab Order. When the user TABS from field to field in the dialog box, Windows will not give the control focus. This is useful for seldom-used data fields. The **Window Designer** will place the SKIP attribute on the control.

Transparent Specify whether you wish the control background to be **Transparent**. This instructs Windows to suppress the rectangular region around the text--the background. Normally, Windows will paint this the same uniform color as the window below the control. This adds the TRN attribute.

Options

You may toggle the following options on or off independently.

Flat Set to True to give your text box a "flat" appearance (the text box control is on the same level as the window, and not recessed).

Required **S**pecifies that the control may not be left blank or zero.

ReadOnly Prevents data entry in this control. Use this to declare display-only data.

Single Specifies the control is only for single line data entry. This is specifically to allow use of TEXT controls instead of ENTRY controls for data entry in languages that write from right to left (such as Hebrew or Arabic).

Resize **Reports ONLY**. Allows the control to increase its height (and the parent report band's height) at runtime to accommodate varying amounts of text from its USE variable.

Position

Lets you set the location and size of the control.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the position and size visually by dragging with the mouse in the **Window Designer**.

To set the location of the control's **Top Left Corner**, set an **X** (horizontal) and **Y** (vertical) coordinate in Dialog units. The coordinate is relative to the top left corner of the structure containing it, that is, the window or the report band.

To set the control's size, choose from the following options for **Width** and **Height**.

Default The Clarion runtime library determines the size of the control based on the applicable font and picture token.

Full The control is the full height or width of the window or report.

Value Lets you set a precise width or height in Dialog units on a window, or in thousandths of inch, points, or millimeters on a report.

Scroll Bars

Horizontal To add a horizontal scroll bar to the control, set the **Horizontal** property to True. Scroll bars only appear when the contents of the text box are bigger than the window.

Vertical To add a vertical scroll bar, set the **Vertical** property to True.

These options add the HSCROLL, VSCROLL, and HVSCROLL attributes to the control.

Quick Links

Embeds Accesses the **Embedded Source** dialog for points surrounding the event handling for this control only.

Toolbar Properties

Lets you edit settings for your TOOLBAR.

Note:

The Property View presents you with two ways to viewing a control's properties.

At the top of the Property View, press the categorized button  to view a control's properties by category (the deafult).

Or press the alphabetical button  to view a control's properties in alphabetical order.

This help documents displays the properties in the categorical view.

Color

Enter a valid color equate in the **TextColor** or **Background** fields, or press the ellipsis (...) button to select a color from the **Color** dialog. The **Window Designer** adds the COLOR attribute to your control declaration.

See ..\LIBSRC\EQUATES.CLW for a list of valid color equates. See *Windows Design Issues* in the *User's Guide* for a discussion on using color to enhance your application.

Design

Locked	"Freezes" the control so that subsequent data dictionary changes are not applied. You can override the #Freeze attribute for all controls or for individual controls. See **Application Options**.

General

Cursor	The *Cursor* field (the CURSOR attribute) lets you specify an alternate shape for the cursor when the user passes the cursor over the control. The drop-down list provides standard cursor choices such as I-Beam and Crosshair. To select an external cursor file (whose extension must be .CUR), choose **Select File** from the drop-down list, then pick the file using the standard file dialog.
Layout	Indicates the orientation of the control. **Left to Right** maintains the original layout specified in the Window Designer. **Right to Left** essentially "flips" the controls' display as a mirror image of the layout specified in the Window Designer. Default field navigation moves from right to left. The setting in the Application Frame will cascade its setting to all child windows and controls that have the **default** setting active.
NoMerge	Specifies *not* to merge an MDI tool bar into an application frame tool bar at run time.
TextFont	Calls the **Select Font** dialog which lets you select the font (typeface), size, style (such as bold or italic), color, and font effects (underline and strikeout) for the selected control or window. As you choose options, the dialog box displays a sample of the selected font.
Use	This defines the USE attribute for the control. Type a field equate label to reference the control in executable code, or the name of a variable.

Wallpaper-BackgroundImage	To provide a background image for the window's client area, specify an image filename. Type the filename or press the ellipsis button (...) to select a file. See WALLPAPER in the *Language Reference*.
Mode	Specify how the window displays the background image. Choose from:
Stretched	The image expands to fill the entire client area.
Centered	The image displays at its default size and is centered in the window's client area.
Tiled	The image displays at its default size and is repeated so it fills the entire client area.

Position

Lets you set the size of the toolbar. The toolbar is always located at the upper left corner of the window, that is, any X and Y coordinates are ignored.

The **Position** tab lets you specify the AT attribute. Filling in the attribute manually is optional--you may set the height visually by dragging with the mouse in the **Window Designer**.

To set the toolbar's size, choose from the following options for **Width** and **Height**.

Default	The Clarion runtime library determines the size of the toolbar, about twice the height of the window's title bar and the full width of the window.
Value	Lets you set a precise width or height in Dialog units.

If the toolbar's width is less than the width of the window, the **Window Designer** displays a full width toolbar to remind you that the "unused" space to the right of the toolbar may be used when merging toolbars.

Quick Links

Embeds	Accesses the **Embedded Source** dialog for this procedure.

Index