

Tutorial Exercise 1

In this exercise, you'll create a new database dictionary, set several preformatting options, define a relationship and set Referential Integrity constraints, then use the Application Wizard to generate and run either a 32-bit or 16-bit application depending on your operating system. The applications will access a TopSpeed data file that's been set up and "pre-loaded" with data for you.

Rapid Application Development

You should be able to do this within 20 minutes.

- Can you create an application faster than with your current development tools?
- Yes. This section shows you how.

The Data

We'll take two tables which are part of a larger Order/Entry database, the whole of which includes five files. These tables reside in one physical file, in the \CWEVAL\TUTOR01 directory.

The data file is in the TopSpeed file format. The TopSpeed file format is one of the so-called "superfile" formats: it can include multiple tables (and keys) residing in a single physical DOS file. It also automatically compresses memo fields.

The file already contains data, so that when you run the application, you'll be able to "play with" some existing data.

Skills

In this exercise, you'll learn to:

- Import file definitions from an existing file.** You'll be able to use this skill to create applications immediately from databases that already exist on your hard drive or network.
- Preformat controls by field.** You can specify that a particular database field should always have a special control (such as an option box with radio buttons) or a special font and font style for an entry-type control. Since you can use a database dictionary to create many applications that maintain the same data, this not only saves time, but provides a common look and feel for the applications that maintain the same database.

Skills

You'll also learn to:

- Define Relationships**
- Define Referential Integrity Constraints.**
- Run the Application Wizard.**
- Define the target operating system.** (You'll create a 32-bit application if you run Windows 95 or Windows NT).

Reminder

We know it's difficult to switch back and forth between the development environment and this document. That's why we've provided another document with all these instructions, formatted to letter sized paper, ready to print. Open the document called D:\DOC\TUTOR01.PDF (where D: is your CD-ROM drive letter) by double-clicking it in Explorer or File Manager, then print it.

Alternatively, if you're viewing this in full screen mode, press ESC to reduce it to a regular window, so that you can ALT-TAB between this document and the Clarion for Windows development environment.

Start the Development Environment

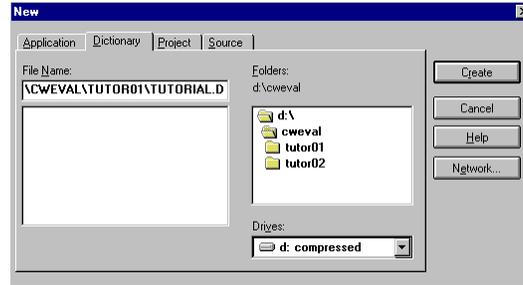
If the development environment is not already running, open it by choosing it from the Start menu, or clicking on its Program Manager icon. It should look like this.



Create a New Dictionary -1-

- From the development environment menu, choose File ► New. In the New dialog, click on the Dictionary tab. The applications you create with Clarion for Windows don't have to be database applications. But when they are, the best starting point is the data dictionary.

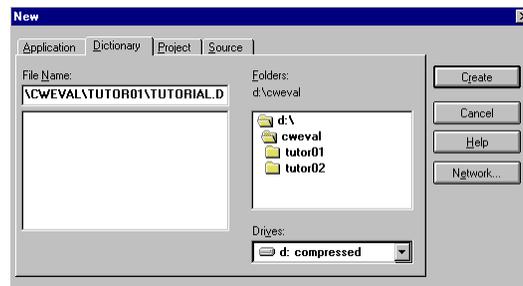
It's important that you follow the directions on the next "page," so that you create your dictionary in the subdirectory in which we've placed your data files.



Create a New Dictionary -2-

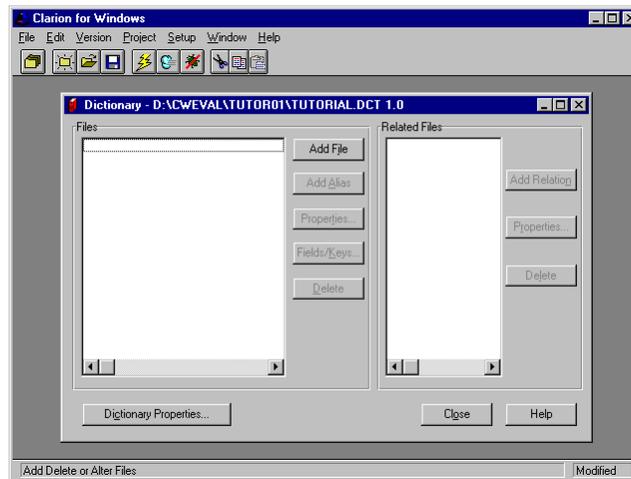
- Type \\CWEVAL\\TUTOR01\\TUTORIAL.DCT in the File Name box of the New dialog. Then press the Create button.

You can optionally locate the directory called \\CWEVAL\\TUTOR01 in the folders list, walking the directory tree as necessary by double-clicking folders; then type in the file name. It's important to specify the correct subdirectory, because we already placed a data file for your application there.



Import a File Definition -1-

At this point the Dictionary dialog appears, with your dictionary file name (\\CWEVAL\\TUTOR01\\TUTORIAL.DCT) at the top. The list at the left holds file or table names. The list at the right will hold the relationships you define for those files.



Import a File Definition -2-

- From the development environment menu, choose File ► Import File.

The Select File Driver dialog includes a dropdown list showing the database drivers installed in your system. You'll choose TopSpeed, which is the default. But you can take a moment to drop down the list to view the other drivers available.

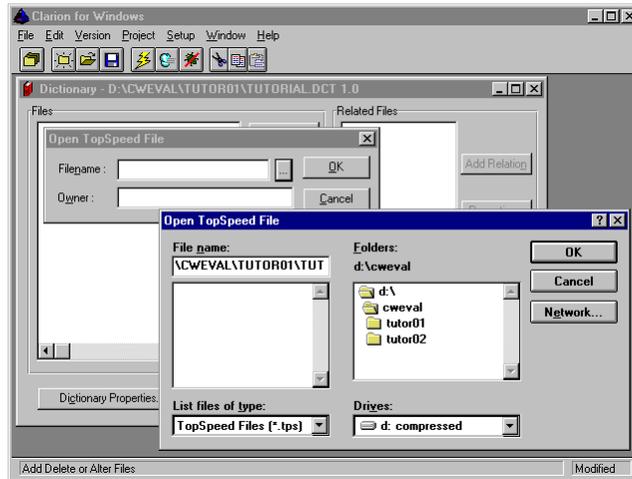
Import a File Definition -3-

- In the Select File Driver dialog, choose the TOPSPEED File Driver, then press the OK button.



Import a File Definition -4-

- In the Open TopSpeed File dialog, press the ellipsis (...) button, then choose \CWEVAL\TUTOR01\TUTORIAL.TPS in the file dialog which then appears. Then press the OK button to close the Open TopSpeed File dialog



Import a File Definition -5-

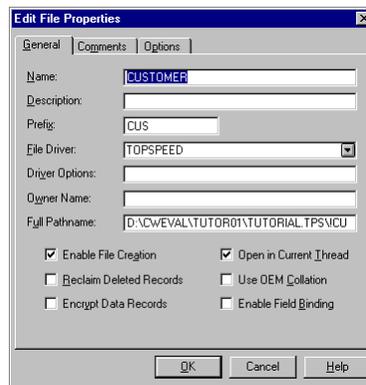
Since the file contains multiple tables, you need to specify the table you want.

- In the Select TopSpeed table dialog, choose Customer (it's the default, since it's the first table listed, and press the OK button.



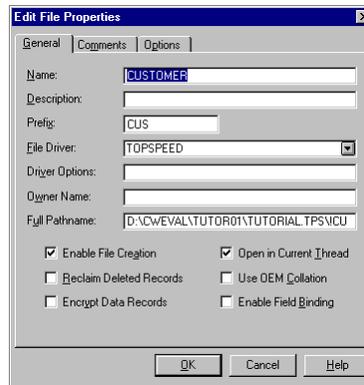
Import a File Definition -6-

The Edit File Properties dialog appears. It allows you to specify options for the FILE structure as a whole. In this case, the default options specify that the application should automatically create a new, empty file if none exists, and that each thread should have its own separate record buffer.



Import a File Definition -7-

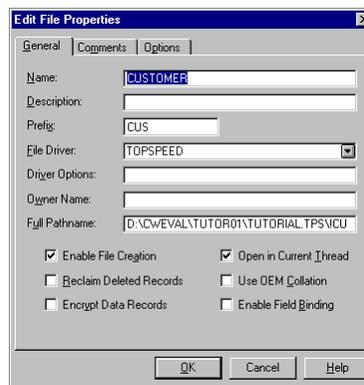
Also, for your convenience, the Comments tab provides a text box for notes; so when you're trying to remember how you designed your database six months down the road, you can read your own explanation!



Import a File Definition -8

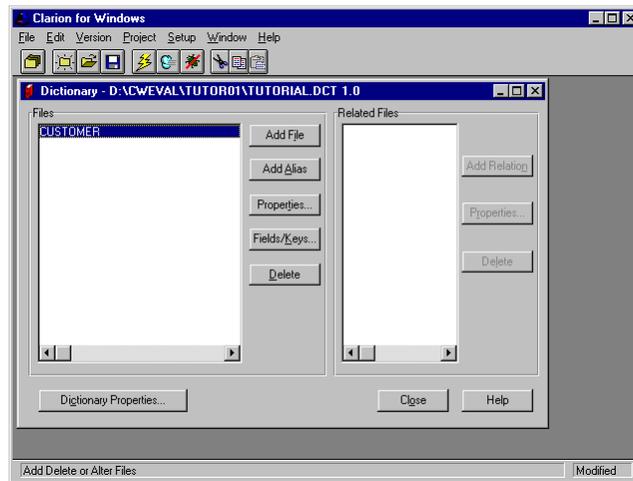
The dictionary editor also allows you to set checkpoints, so that you can “undo” a revision should you run into a problem.

- Press OK to close the Edit File Properties dialog.



The Dictionary So Far

So far, the dictionary holds one file definition. That includes all the field and key definitions.



Import Second File Definition -1-

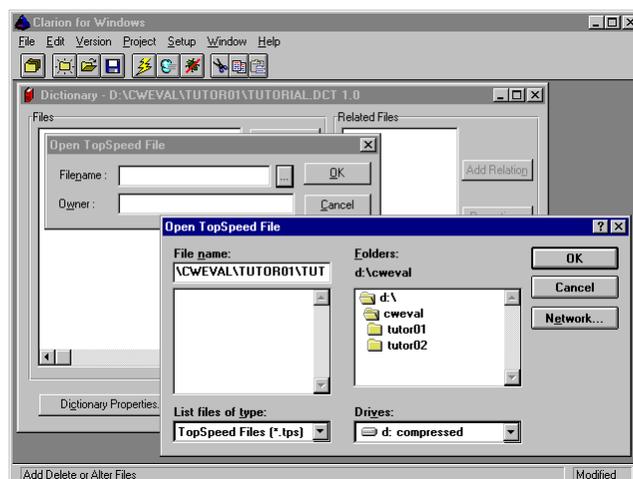
Next, you'll add a second, related file definition. It will be the Orders table, which is related to the Customer table by the customer number.

- From the development environment menu, choose File ► Import File.
- In the Select File Driver dialog, choose the TOPSPEED File Driver, then press the OK button.



Import Second File Definition -2-

- In the Open TopSpeed File dialog, press the ellipsis (...) button, then choose \CWEVAL\tUTOR01\tUTORIAL.TPS in the file dialog which then appears. Then press the OK button to close the Open TopSpeed File dialog.



Import Second File Definition -3-

Since the file contains multiple tables, you need to specify the table you want.

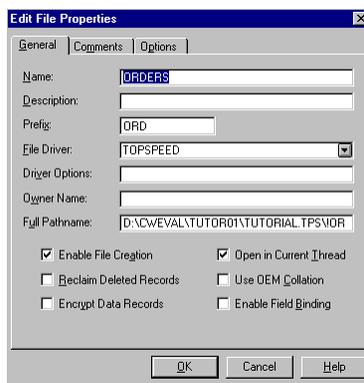
- ❑ In the Select TopSpeed table dialog, choose Orders, and press the OK button.



Import Second File Definition -4-

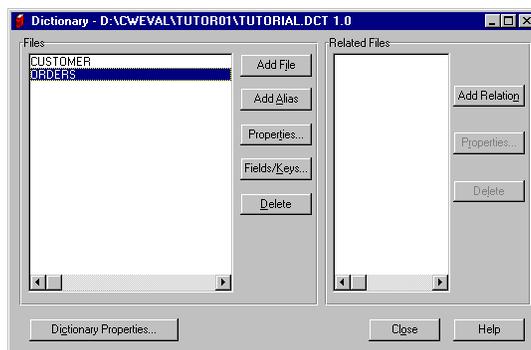
The Edit File Properties dialog appears. Accept the defaults for the Orders table.

- ❑ Press OK to close the Edit File Properties dialog.



The Dictionary So Far

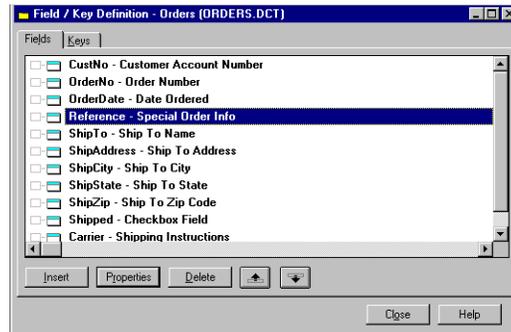
Now, the dictionary holds two file definitions. That includes all the field and key definitions.



Set Auto-Increase Keys -1-

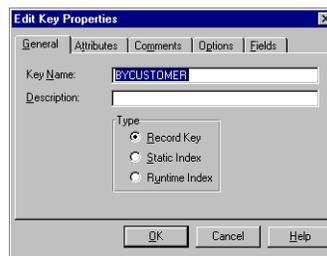
Key attributes include primary, exclude null, and others. You'll set the customer number key to auto-number, which tells the Application Generator you want to generate code to auto-increment the key value for any new records added to the database.

- With the Orders file selected, press the Field/Keys button, opening the Fields/Key Definition dialog.



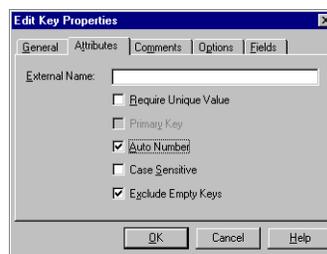
Set Auto-Increase Keys -2-

- Select the Keys tab. The BYCUSTOMER key should be selected.
- Press the Properties button. The Edit Key Properties dialog appears.



Set Auto-Increase Keys -3-

- Select the Attributes tab in the Edit Key Properties dialog.
- Check the Auto Number box.



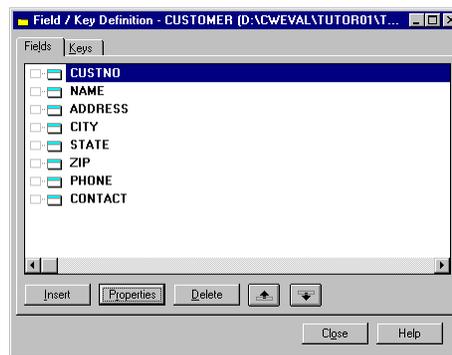
Set Auto-Increase Keys -4-

- Press the OK button to close the Edit Key Properties dialog.
- Press the Close button to close the Field/Key Definition dialog. This returns you to the Dictionary dialog.

Set Auto-Increase Keys -5-

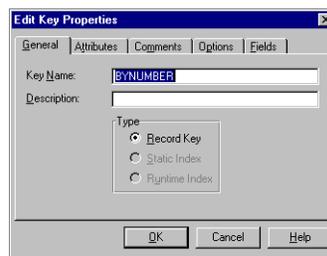
Now you need to set the Auto-Number attribute for the Customer file.

- Select the Customer file in the Dictionary dialog.
- Press the Fields/Keys button. This displays the Field/Key Definition dialog.



Set Auto-Increase Keys -6-

- Select the Keys tab. The BYNUMBER key should be selected.
- Press the Properties button. The Edit Key Properties dialog appears.



Set Auto-Increase Keys -7-

- Select the Attributes tab in the Edit Key Properties dialog.

- Check the Auto Number box.



Set Auto-Increase Keys -8-

- Press the OK button to close the Edit Key Properties dialog.
- Press the Close button to close the Field/Key Definition dialog. This returns you to the Dictionary dialog.
- Choose File ► Save, to save your work so far.

Pre-Formatting Controls

You can specify that a particular database field should always have a special control (such as an option box with radio buttons) or a special font and font style for an entry-type control. Since you can use a database dictionary to create many applications that maintain the same data, this not only saves time, but provides a common look and feel for the applications that maintain the same database.

- Which development environment supports greater code reusability?
 - ✓ Clarion. By storing application options in the database dictionary, you have a head start on all projects that reference the same database.

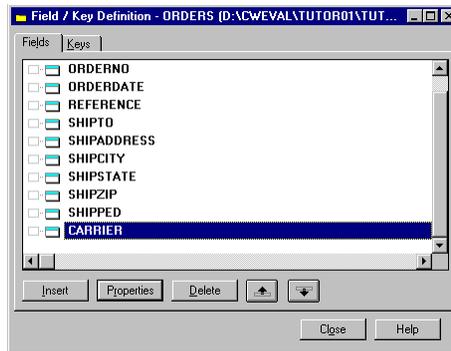
Pre-Formatting A Control

The Orders file has a field called Carrier, which holds a string value describing how the order should be shipped. We'll pre-define this field in the dictionary, so that the Application Wizard knows that we want to create radio buttons that offer the end user a choice of "Mail," "UPS," or "Other." We'll set the default value as "Mail."



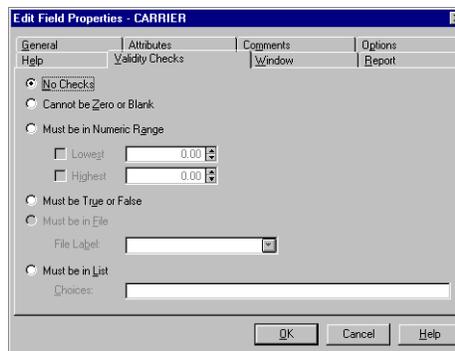
Pre-Formatting the Carrier Field -1-

- Select the Orders file in the Dictionary dialog.
- Press the Fields/Keys button. This displays the Field/Key Definition dialog.
- Select the Carrier field.



Pre-Formatting the Carrier Field -2-

- With the Carrier field selected, press the Properties button. The Edit Field Properties dialog appears. This is the dialog for storing all the information about a given field. The General tab sets the basic properties, including the field name and data type. This is a one-byte field.
- Click on the Validity Checks tab.



Pre-Formatting the Carrier Field -3-

You'll take advantage of a shortcut. When you define a list of allowable values in the Validity Checks, the Dictionary Editor automatically pre-formats the control as an option box with radio buttons.

Additionally, you'll take advantage of another feature to save space in the data file. For a one-byte field, your application will automatically store only the first character of the values you define in the list (of course, you must be sure that they're unique) in the data file.

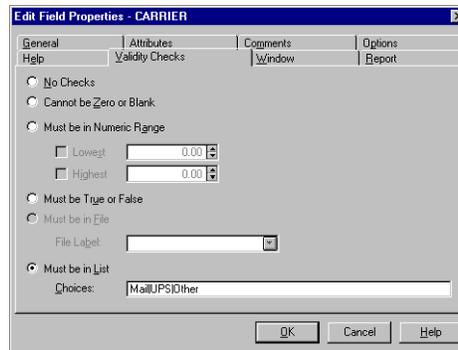
- ❑ Select the Must be in List radio button on the Validity Checks sheet.

Pre-Formatting the Carrier Field -4-

- ❑ Type the following in the Choices box, including the pipe symbols, which separate the valid choice values:

Mail|UPS|Other

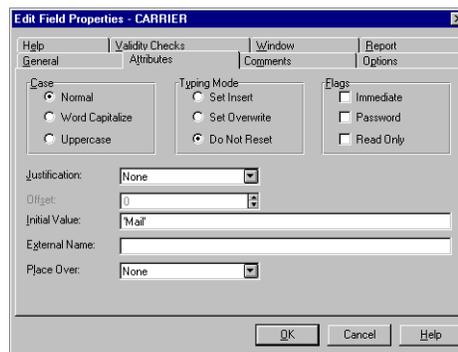
“No selection” is a valid radio button choice, and the end user will see exactly that for each new record unless you specify otherwise. You **do** want to set a default value for the carrier type.



Pre-Formatting the Carrier Field -5-

- ❑ Click on the Attributes tab.
- ❑ Type the following in the Initial Values box. Be sure to enclose the word between single quotes:

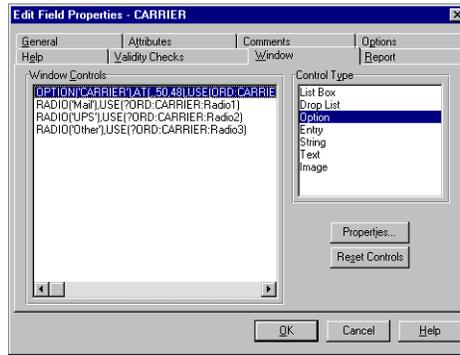
'Mail'



Pre-Formatting the Carrier Field -6-

- ❑ You'll just check your work. Click on the Window tab. This stores your control choices for the update form that the Application Wizard will generate for you.

It should look like the illustration.



Pre-Formatting the Carrier Field -7-

Notice that you could also have specified a drop down list box, or various other entry controls. If you were to press the Properties button on this sheet, you'd find additional options, such as the ability to specify the font, font size, and font style for the control. For now, don't change the options.

- Press the OK button to close the Edit Field Properties dialog.
- Press the Close button to close the Field/Key definition dialog.
- Choose File ► Save, to save your work so far.

Pre-Formatting a Field Picture

A field picture formats a value for a window or report control. For example, you can store a long value in a database with a value of 2710 hex (10,000 decimal). A currency picture might format the value so that it appears in an entry control as \$10,000.00, to cite only one example. A numeric picture could format it to appear as 10000, 10,000, or maybe even 00000000000000000000000010000, and others, depending on the format picture you choose.

You'll use a pattern picture to format a number stored in the phone field. In the data file, it's stored in the format 2125551212. You'll format it to appear to the end user as (212) 555-1212.

Pre-Formatting the Phone Field -1-

- Select the Customer file in the Dictionary dialog.
- Press the Fields/Keys button. This displays the Field/Key Definition dialog.
- Select the Phone field.

The following topics will define the relationship, and set the Referential Integrity constraints. Referential Integrity is the means by which the database relationships are maintained. Verifying unique values for the fields comprising the primary key, and excluding null values are part of this process. Cascading a key value change in a parent record to its related children, or deleting the child records when a parent is deleted are also part of the process.

Referential Integrity Constraints

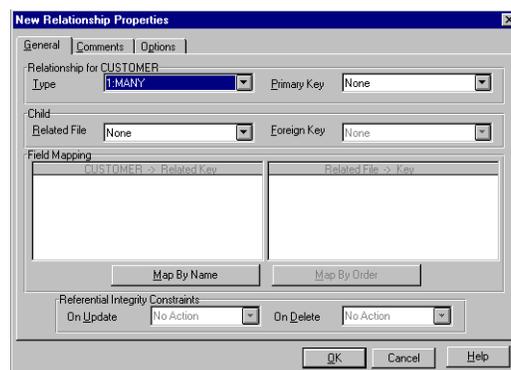
Clarion implements RI checks in the generated code. This allows you to support RI for **any** database. You can also specify no RI checks, and let the DBMS handle it for you, where applicable. This option is very convenient, for example, for a company with an older AS/400 database. The older software didn't support RI; so you can have your Clarion application generate the code. If you upgrade the DBMS software to a newer version in which the AS/400 **does** handle the RI, you simply "uncheck" the RI options in your Clarion app, regenerate, and recompile.

Referential Integrity Constraints

- Which development environment has the strongest data dictionary?
 - ✓ Clarion. It's flexible, and the development environment maintains live links between it and your application files. Your applications adapt as your business needs change. All you need to do is update the data dictionary, regenerate the application, and recompile.

Setting the Relationship -1-

- Select the Customer file in the Dictionary dialog.
- Press the Add Relation button. The New Relationship Properties dialog appears.



Setting the Relationship -2-

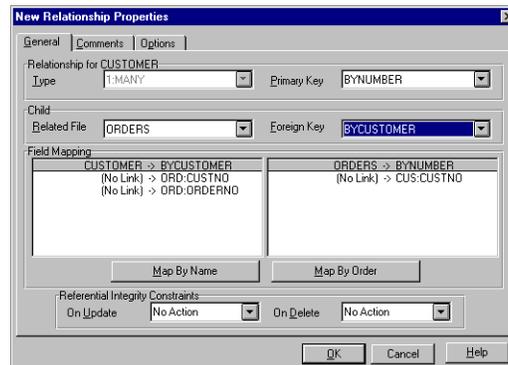
The Customer file appears at the top of the New Relationship Properties dialog.

- Select the BYNUMBER key in the Primary Key drop-down list.

- Select ORDERS in the Related File drop-down list.
- Select BYCUSTOMER from the Foreign Key drop-down list.

Setting the Relationship -3-

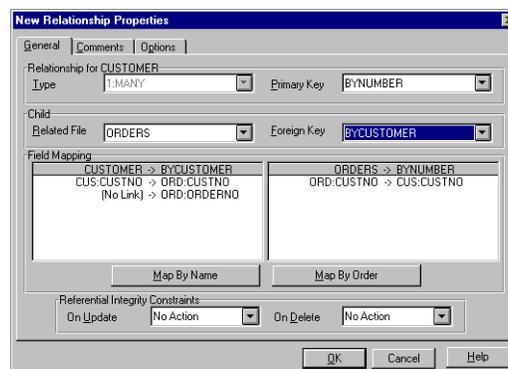
This is how the New Relationship Properties dialog looks so far.



Setting the Relationship -4-

The Orders file key has two components, while the Customer file key has one. You don't have to worry about it; the generated code will take care of everything for you.

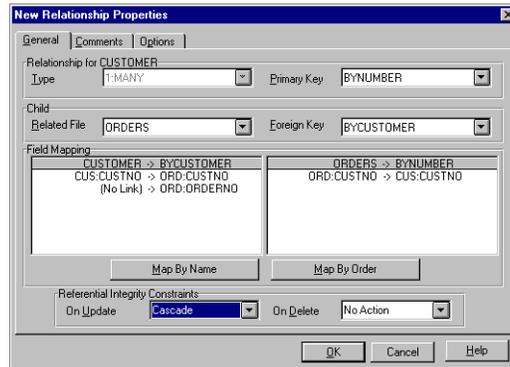
- Press the Map By Name button to define the relationship by field. This matches the customer number fields in the Customer and Orders tables.



Setting the Relationship -5-

Set the RI constraint options. You'll choose Cascade on Update. This would update child records when you update the key value for a parent record. In this case, if you change the customer number in the customer file, it would update the number in the Order file.

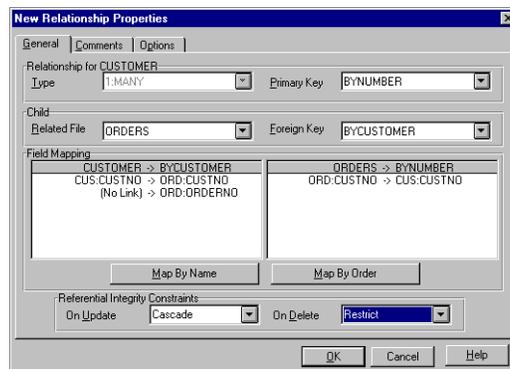
- Select Cascade from the On Update drop-down list.



Setting the Relationship -6-

You'll choose Restrict on Delete. This would disallow the deletion of a parent record if there are related children. In this case, if the end user attempts to delete a customer record with related orders, the application would disallow it.

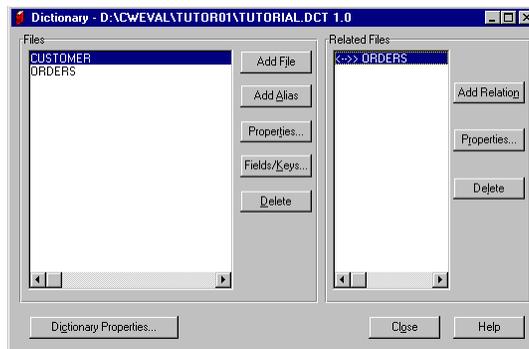
- Select Restrict from the On Delete drop-down list.



Close the Dictionary -1-

- Press the OK button to close the New Relationship Properties dialog.

You're done with the dictionary. You've imported definitions from existing data files, pre-formatted controls, and set RI options. The dictionary should look like this:



Close the Dictionary -2-

- Choose File ► Save, to save your dictionary.
- Press the Close button to close the Dictionary dialog.

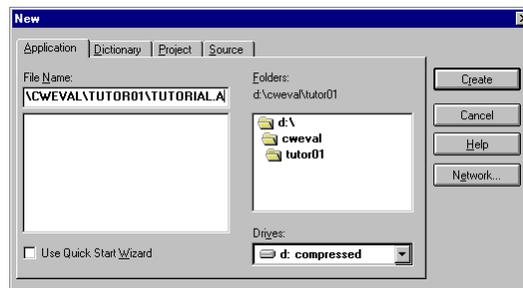
The Application Wizard

The Application Wizard will read your data dictionary and create an application for its maintenance. It will include browses for navigating the file, update forms, and reports.

- Choose File ► New from the development environment menu.
- If not already selected, click on the Application tab.

New Application

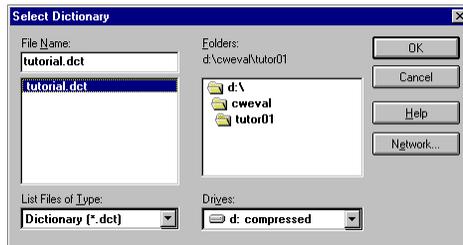
- Be sure that the Use Quick Start Wizard check box is **unchecked**. Quick Start is a convenience feature that lays out a data table quickly; but since we imported existing files, we don't need it for this example.
- Type \CWEVAL\TUTOR01\TUTORIAL.APP in the File Name box.
- Press the Create button.



The Application Properties Dialog -1-

This dialog specifies the name of the .APP file, which stores your application description. You must specify the data dictionary name.

- First make sure the Application Wizard check box is **checked**.
- Press the ellipsis button (...) next to the Dictionary File box.
- Choose \CWEVAL\TUTOR01\TUTORIAL.DCT from the Select Dictionary dialog, then press the OK button to close the Select Dictionary dialog.



The Application Properties Dialog -2-

You're now ready to create your application file. The Application Properties dialog should look something like this.

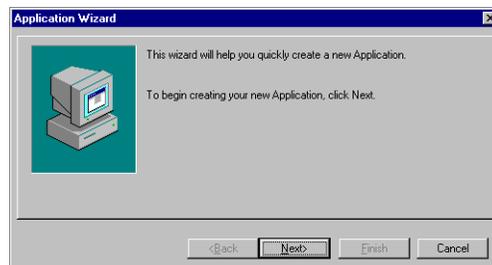


- Press the OK button.

The Application Wizard -1-

The Application Wizard appears. The first sheet is an "intro."

- Press the Next button.



The Application Wizard -2-

The second sheet asks you whether you want the Application Wizard to process all the files/tables in your dictionary; i.e., prepare a browse/form/report for each one.

- Press the Next button; there are only two files in the dictionary, so you want to process them both.



The Application Wizard -3-

The third sheet asks whether you want the App Wizard to overwrite existing procedures in your application. This allows you to run the Application Wizard **after** you've done some work in an application. Unlike one-time-only, one-way wizards in other RAD tools, you can run Clarion wizards any time.

- Press the Finish button. You don't have to worry about anything here.



The Application Wizard -4-

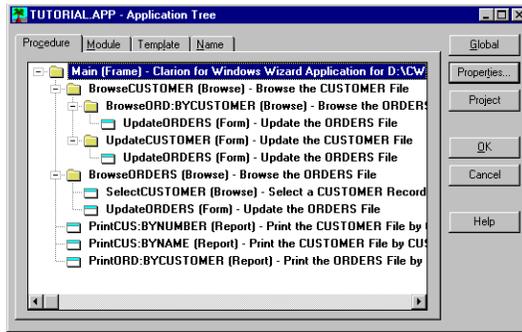
The Wizard Code Generation Progress appears. The Application Generator is reading your data dictionary, and choosing appropriate templates from the template registry.

- Can you create an application faster than with your current development tools?
 - ✓ Yes. You just created a complex application from scratch.



The Application Tree

Your application is ready. You now see the Clarion Application Tree. It's a logical procedure call tree. It organizes your project in a hierarchy of procedures. A window or a report structure can comprise a procedure. Likewise, a source code function can also be a procedure. You can see at a glance how everything is connected. It's hard to misplace your code (the Visual Basic "hidden behind a thousand doors" syndrome).



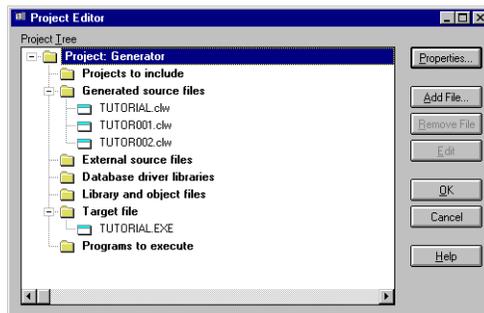
Project Settings -1-

The default target Operating System for the compiler in the Evaluation Edition is 16-bit Windows 3.1. If you're using Windows 95 or Windows NT, we'll change it with a couple of clicks.

If you're using Windows 3.1, [jump ahead](#) five topics.

Project Settings -2-

- Press the Project button. The Project Editor dialog appears, with the top level folder selected. The Project System stores the various compile options and pragma.
- With the top level folder selected, press the Properties button in the Project Editor dialog.

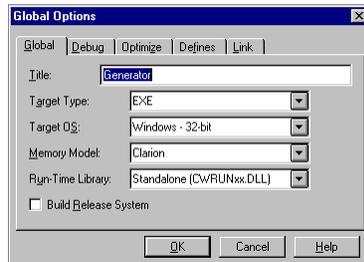


Target OS -1-

- In the Global Options dialog, choose Windows - 32 bit from the Target OS drop-down list. This specifies you want to compile an application to run on Windows 95 or Windows NT.
- The full edition of Clarion for Windows allows you to compile everything into one single executable file. The Evaluation Edition default—Standalone—requires that CWRUNxx.DLL be present when the end user runs your app. Your executables will dynamically link to many functions in the .DLL at runtime. This option is actually helpful for settings where you expect your end user to have many Clarion-created applications on the hard drive; it can literally save megabytes of disk space.

Target OS -2-

This is how the Global Options dialog should look:

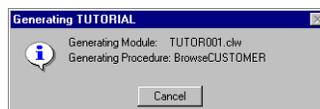


Get Ready to Compile

- Press the OK button to close the Global Options dialog.
- Press the OK button to close the Project Editor dialog.

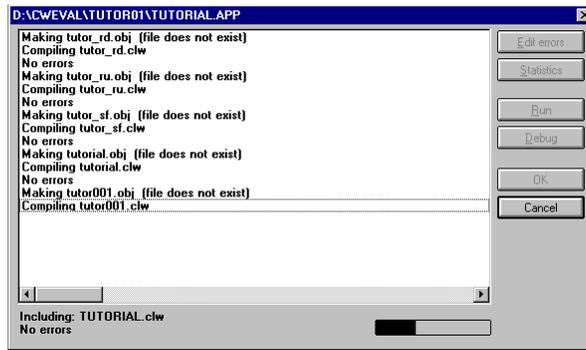
Compile and Run -1-

- Choose File ► Save to save your work so far.
- Press the Make & Run button (sixth tool bar button from the left on the toolbar). It's a blue puff of smoke because Clarion apps leave others in the dust. A progress window reports the progress of code generation.



Compiling

After the Application Generator generates the Clarion language source code, it's converted to an intermediate symbolic language, which in turn is sent to the back-end TopSpeed compiler, which compiles it into the executable.



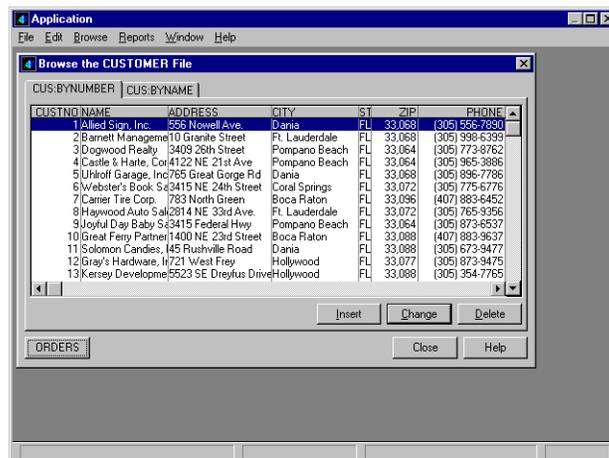
Running the App -1-

Your app should run once the compile process is over. (Note for Windows 95 users, just in case: if you receive a DDE timeout error, you need to make more memory available for disk swapping. The 32-bit compiler needs a lot of memory).

- Choose Browse ► Browse the Customer File from the menu. (Note: the Application Generator picks up the menu text from the description in the dictionary. You only scratched the surface of the pre-formatting options when you edited the dictionary.)

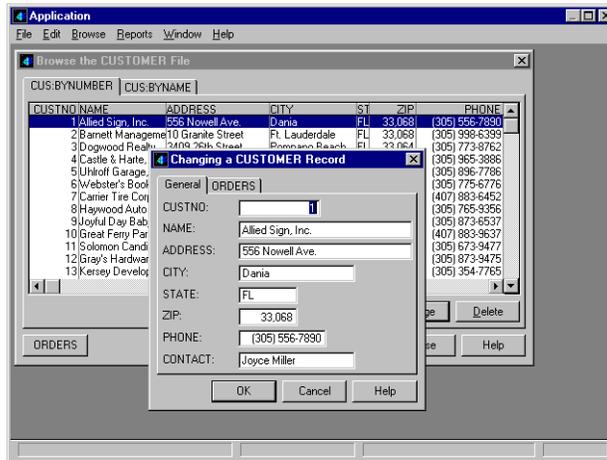
Running the App -2-

Note the formatting for the phone field reflects the pattern picture you stored in the dictionary.



Running the App -3-

- With customer number 1 selected, press the Change button. An update form for the Customer record appears.



Running the App -4-

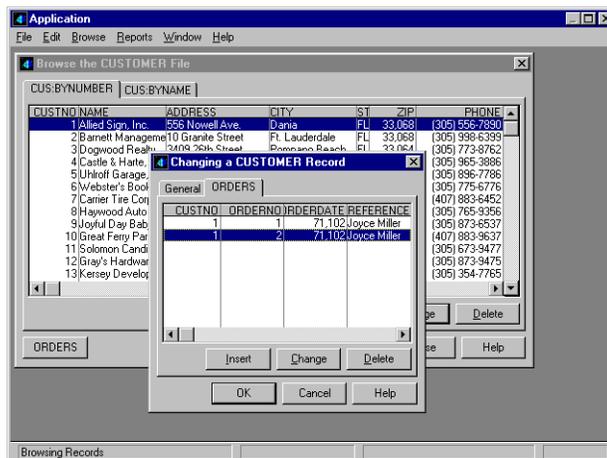
Note the zip code was formatted as `##,###`. That's the default format for a decimal value. You could have specified a picture of `@N05`, which would have formatted it without the commas, and with leading zeroes. In the next exercise, you'll work with a dictionary with all fields pre-formatted. Additionally, descriptions are provided for files and keys, so that the menu items and tabs are also pre-formatted.

The code for this update form includes support for concurrency checking in a networked environment.

Running the App -5-

- ❑ Click on the ORDERS tab in the Changing a Customer Record dialog.

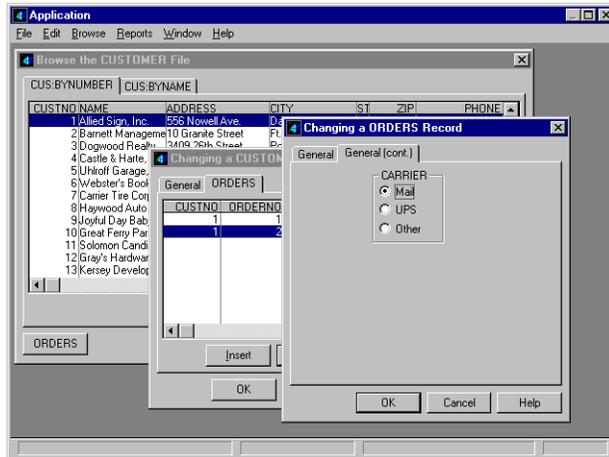
Notice that the Application Wizard automatically provided a listbox showing all the child order records for this customer record.



Running the App -6-

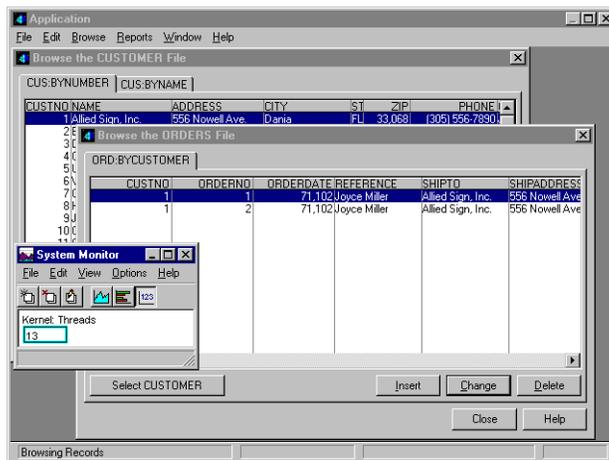
- ❑ With the Orders tab selected, press the Change button in the Changing a Customer Record dialog.

- ❑ Select the General (cont.) tab in the Changing a Orders Record dialog. Notice the option box and radio buttons for the Carrier field; that's the field you pre-formatted.



Running the App -7-

Continue experimenting with the application as you wish. If you set the target OS to 32-bit, you can run the System Performance Monitor to confirm that each browse opens a new thread.



Summary

To summarize, you've just created a multi-threaded app that maintains two related data files, providing visual "links" between them in the windows that the end user sees. You pre-formatted a couple of fields in the data dictionary; those options will migrate to any additional applications you develop from the same dictionary, so imagine how much work you save by preformatting all the fields that need it (e.g., the zip code field).

Where to Go From Here

You can either go on to the next exercise, or take a shot at creating an application from your own data files. Be sure to work with a copy of your database first, until you're more familiar with Clarion.

Do you have dBase or Clipper files? You can use the direct drivers. For Microsoft Access files, you can use an ODBC driver (note: you cannot use the Microsoft Office 4.x Access driver—it's designed **only** for use by Microsoft Office! Be sure your Access ODBC driver—ODBCJT16.DLL or ODBCJT32.DLL—is version 2.00.23.17 or higher. Select the driver in File Manager or Explorer and choose File/Properties).

The Next Exercise

The next part of the exercise takes a dictionary with more extensive formatting (based on these two tables, plus the others from the same database), and introduces you to the template interface. You'll customize some of the windows that the Application Wizard creates for you, adding even more functionality.

Final Note

- ❑ Which development environment gives you the overall speed and flexibility you need to create the best solutions to a wide array of your end users' needs?
 - ✓ Clarion. You can immediately create an application to manage a database with an unlimited number of tables, just from this short introduction. Remember that we suggest you work with a copy of your data, until you learn a little more about Clarion, or better still, go on to exercise two!

Tutorial Exercise 2

In this exercise, you'll work with the template interface.

We'll work with the database from exercise one, which contains the fields necessary for an Order/Entry application. In that exercise, the Application Wizard built all the "basics," according to the database structure.

You can think of the Application Wizard as an enormous "head start" for your development projects. Because once you've run the Wizard, the Application Generator allows you to add further customizations. And you don't have to write a single line of code if you don't want to.

The Customizations

Let's add some "finishing touches" to an application built on our Order/Entry database. We'll add a "line total" for each individual order, and a "grand total" for the line totals.

Clarion templates help you place user interface elements that already know how to implement a complete business solution. You don't need to attach the code that totals the records in the list. The template already contains the code. You just indicate what needs totalling by choosing from a list.

The Files

You'll use a dictionary based on the same database used in exercise 1. This time, however, it will include all the tables (five) and additional pre-formatting. For example, we've added descriptions for the keys. These descriptions are picked up by the Application Wizard, which places them on the tabs that select the sort order in a browse window. In the first exercise, when you saw these tabs, they had semi-cryptic prefix-key names, such as CUS:BYNUMBER.

You'll also work with an application file we've created for you. To save you time, we already generated the .APP file using the Application Wizard. We didn't customize it at all. You will, in this exercise.

Skills

In this exercise, you'll learn to:

- Use the Window Formatter.** You'll resize one control and add another to an existing window.
- Use the Listbox Formatter.** You'll add a new column to a listbox created by the Application Wizard.
- Customize Control Properties via the Template Interface.** You'll edit the actions of a browse listbox so that it places a total for one column in an entry box you'll place just below the list box.

- ❑ **Define a Data Variable.** You'll define two new variables, to keep track of the line total on an individual order, and the grand total for the order.

Reminder

We know it's difficult to switch back and forth between the development environment and this document. That's why we've provided another document with all these instructions, formatted to letter sized paper, ready to print. Open the document called D:\DOC\TUTOR02.PDF (where D: is your CD-ROM drive letter) by double-clicking it in Explorer or File Manager, then print it.

Alternatively, if you're viewing this in full screen mode, press ESC to reduce it to a regular window, so that you can ALT-TAB between this document and the Clarion for Windows development environment.

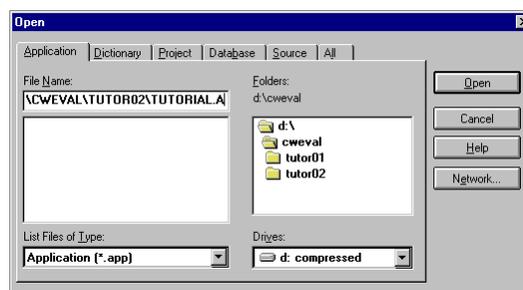
Start the Development Environment

If the development is not already running, open it by choosing it from the Start menu, or clicking on its Program Manager icon. It should look like this.



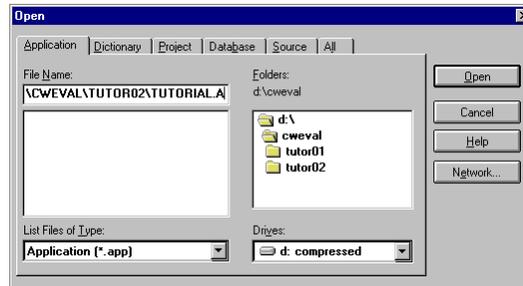
Open the Application File -1-

- ❑ From the development environment menu, choose File ► Open. In the Open dialog, click on the Application tab.
- ❑ Type \CWEVAL\TUTOR02\TUTORIAL.APP in the File Name box of the Open dialog. Then press the Open button.



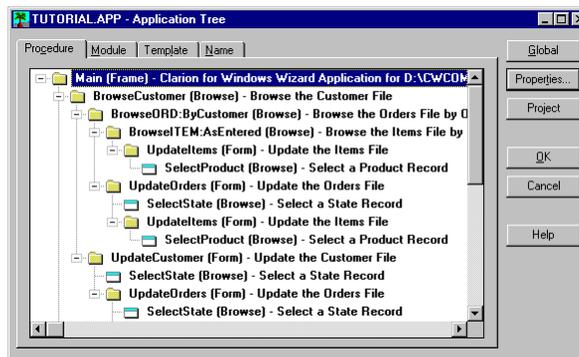
Open the Application File -2-

Note: You can optionally locate the directory called \CWEVAL\TUTOR02 in the folders list, walking the directory tree as necessary by double-clicking folders, and then type in the file name. It's important that you specify the correct subdirectory, because we already placed an application file there.



The Application File

The Application Tree dialog appears. You'll immediately notice a few differences from the previous application. You'll note browse windows, update forms, and reports for three additional tables, Items, Products, and States, in addition to the Customer and Orders tables.



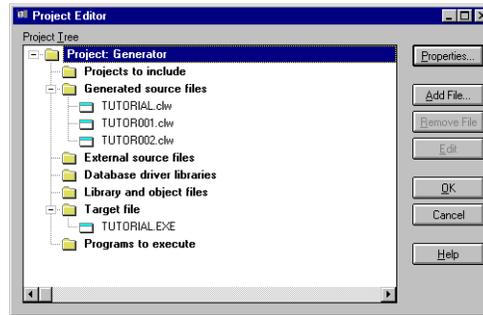
Set a 32-bit Target

As you recall, the default target Operating System for the compiler in the Evaluation Edition is 16-bit Windows 3.1. If you're using Windows 95 or Windows NT, you can change it with a couple of clicks.

If you're using Windows 3.1, [jump ahead](#) four topics.

Project Settings -1-

- Press the Project button. The Project Editor dialog appears, with the top level folder selected. The Project System stores the various compile options and pragma.
- With the top level folder selected, press the Properties button in the Project Editor dialog.



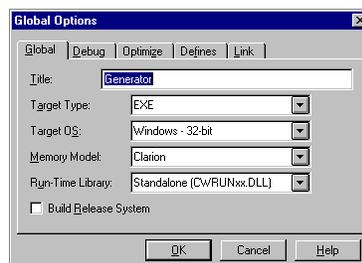
Target OS -2-

- ❑ In the Global Options dialog, choose Windows - 32 bit from the Target OS drop-down list. This specifies you want to compile an application to run on Windows 95 or Windows NT.
- ❑ The full edition of Clarion for Windows allows you compile everything into one single executable file. The Evaluation Edition default—Stand-alone—requires that CWRUNxx.DLL be present when the end user runs your app. Your executables will dynamically link to many functions in the .DLL at runtime. This option is actually helpful for settings where you expect your end user to have many Clarion-created applications on the hard drive; it can literally save megabytes of disk space.

Target OS -3-

This is how the Global Options dialog should look before you close it.

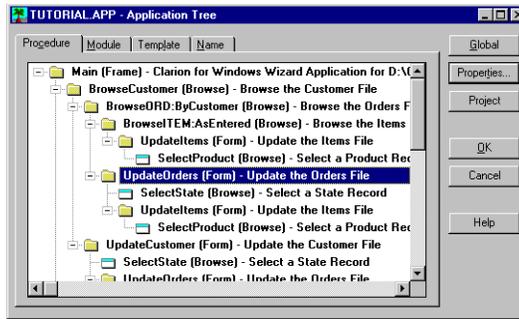
- ❑ Press the OK button to close the dialog box.



Adding a Data Variable

In preparation for adding a total to a browse listbox, you'll define a data variable to hold the running total.

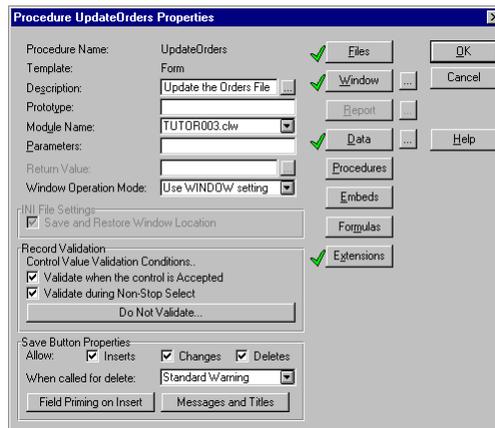
- ❑ Select the Update Orders form in the Application Tree (it should be the seventh procedure from the top, when all the folders are expanded).
- ❑ Press the Properties button.



The Procedure Properties Dialog

The Procedure Properties dialog provides access to all the resources, files, variables, template code, and source code for a procedure. You'll define a new data variable.

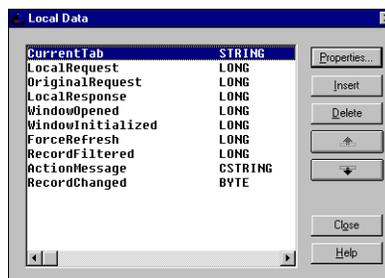
- ❑ Press the Data button.



The Local Data Dialog

The Local Data dialog appears. The template code already defined the variables you see listed. The variable names are self explanatory; for example, CurrentTab is the text on the tab of the sheet currently selected by the end user.

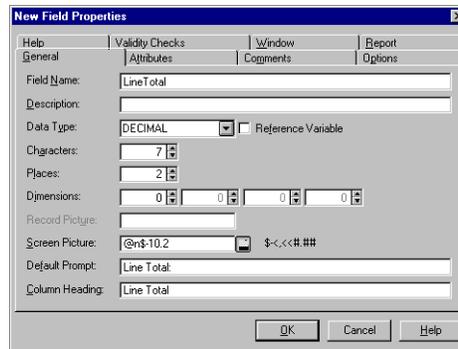
- ❑ Press the Insert button to define a new variable.



The New Field Properties Dialog

Name, type, and pre-format your variable:

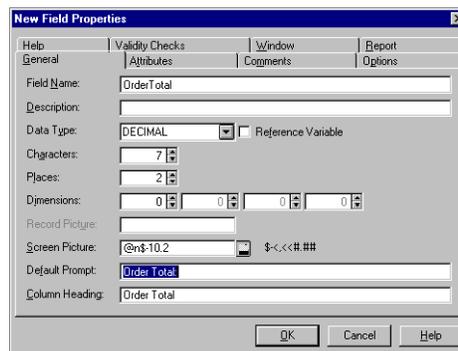
- Type LineTotal (one word) in the Field Name box.
- Choose Decimal from the Data Type list. Since this will hold a currency value, you want to use integer math.
- Type @n\$-10.2 in Screen Picture box. This adds currency formatting for window controls.
- Press OK.



Defining the Second Variable -1-

The New Field Properties dialog reappears, ready to accept a second variable:

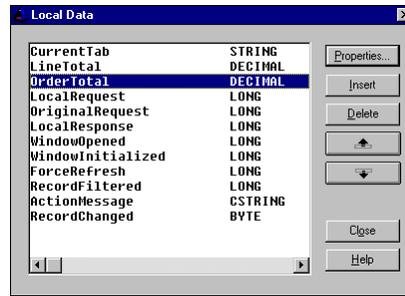
- Type OrderTotal (one word) in FieldName box.
- Choose Decimal from the data type list.
- Type @n\$-10.2 in Screen Picture box.
- Press OK.



Defining the Second Variable -2-

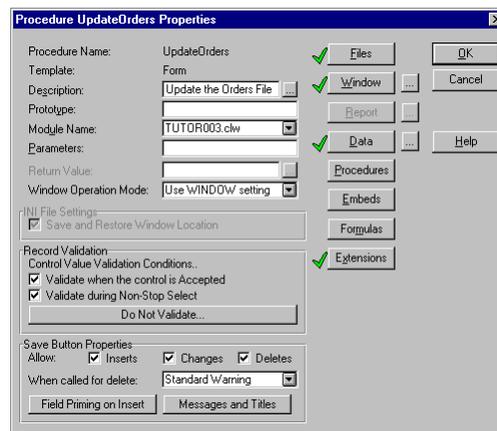
The New Field Properties dialog reappears, ready to accept another variable. Many of the dialogs that allow you to define files, fields or variables cycle to a blank dialog immediately after you define an element, to help you define a series of elements quickly.

- ❑ Press Cancel—you've defined all the variables necessary.



The Procedure Properties Dialog

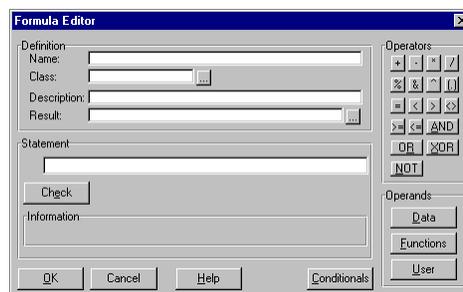
Press Close to return to the Procedure Properties dialog. A check next to the Data button indicates that the procedure includes variables that you've defined.



Defining a Formula

In this section, we'll create a formula to provide a calculated (derived) field in the Items listbox in the "Changing an Order Record" dialog. This shows you the formula editor, which can save you hand coding by automatically constructing Clarion language functions, including conditional structures.

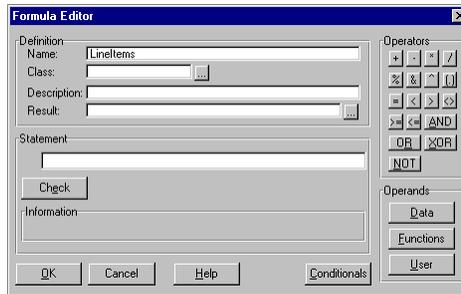
- ❑ Press the Formulas button to open the Formula Editor.



The Formula Editor -1-

You'll create a formula name, using a class that the browse listbox control template already understands how to handle:

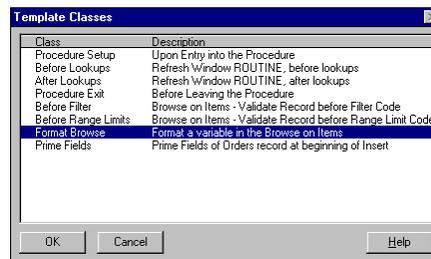
- Type Lineltems (one word) in the Name box.
- Press the ellipsis button (...) next to the Class box.



The Formula Editor -2-

There are several formula classes defined for use in the templates. The Format Browse class calculates a formula, and places the result in a variable for each record read into the browse.

- Select Format Browse.

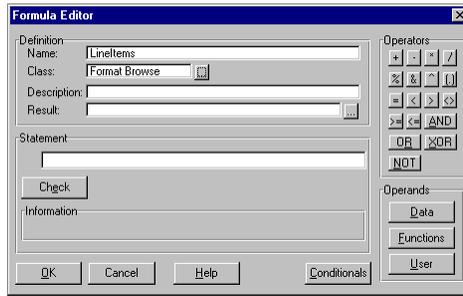


The Formula Editor -3-

- Press OK to close the Template Classes dialog.

Now you can indicate the variable that should hold the formula result for each record read.

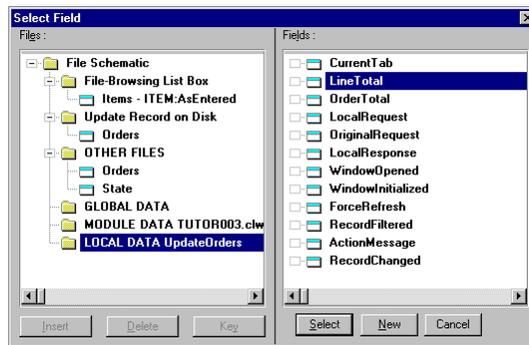
- Press the ellipsis button (...) next to the Result box



The Formula Editor -4-

Choose the variable you previously defined. Notice that you have access to all variables, files and fields in the Select dialog.

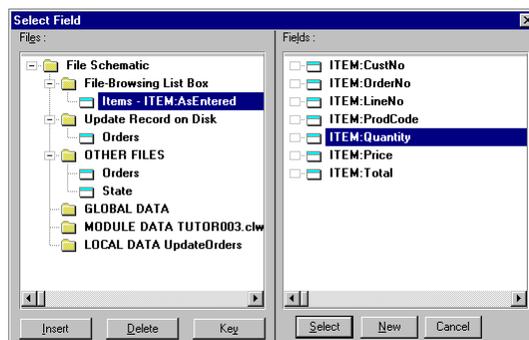
- Select Local Data from the Files list.
- Select LineTotal from the Fields list.
- Press the Select button.



The Formula Editor -5-

Now you can create the formula. The Line Total should be the Quantity of items times the Price per item. You can use the Select dialog to indicate the Quantity and Price, which are fields in the Items file.

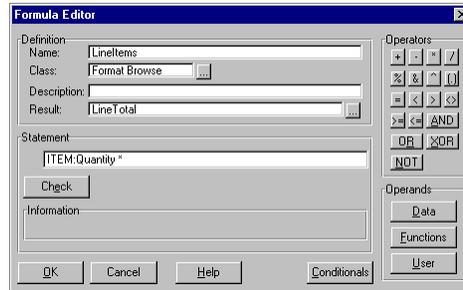
- Press the Data button (in the operands group).
- Select Items from the Files list.
- Select Item:Quantity from the Fields list.
- Press the Select button.



The Formula Editor -5-

You've now chosen one component of the formula. Now you can add the operator:

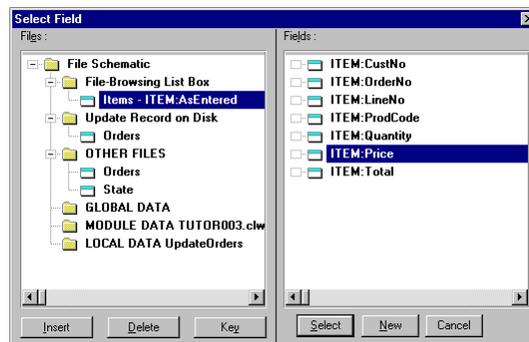
- Type space-asterisk-space.



The Formula Editor -5-

Now you can add the second part of the formula:

- Press the Data button.
- Select Item:Price.
- Press the Select button.

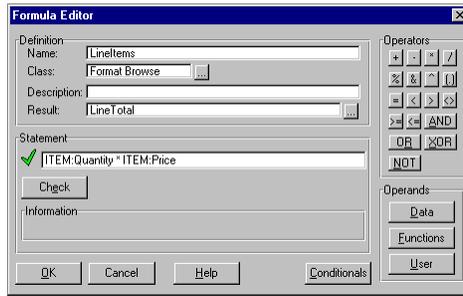


The Formula Editor -6-

Now you can use the Formula Editor to check the syntax of the statement you've constructed.

- Press the Check button.

A checkmark will appear next to the statement, to confirm that the syntax is correct.

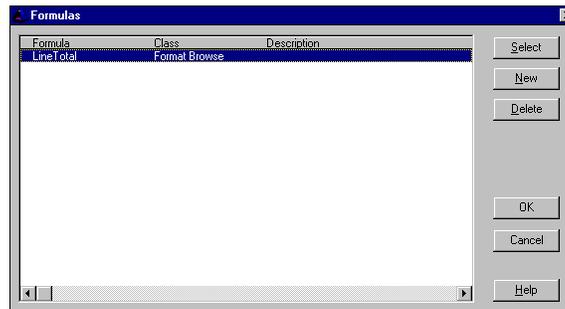


The Formula Editor -7-

- Press OK; you only wish to create the single formula.

You've finished the formula. The Formulas dialog will appear to allow you to enter a second formula.

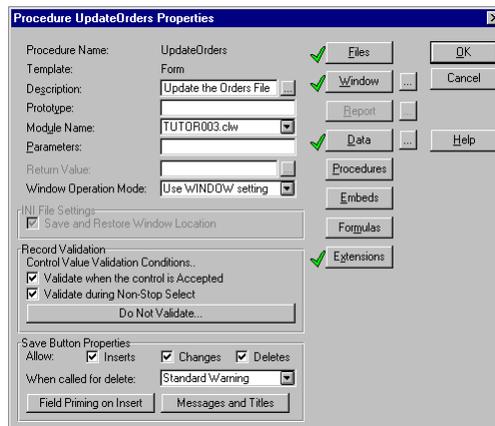
- Press OK.



The Formula Editor -8-

You can now close the Procedure Properties dialog.

- Press OK.
- Choose File ► Save, to save your work so far.



Editing a Window

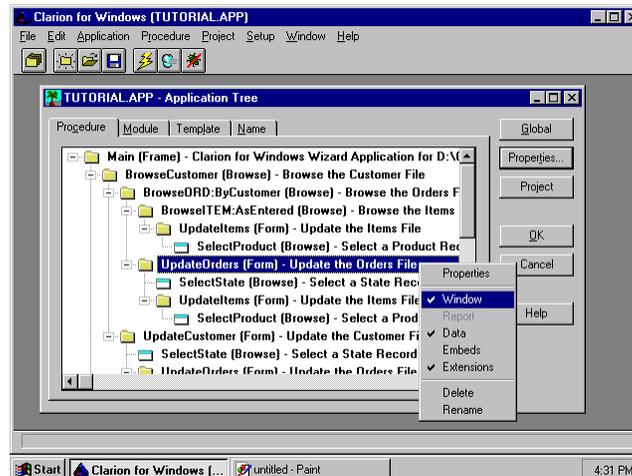
You'll now use one of the visual design tools—the Window Formatter—to edit and add controls to a window.

You'll add a line total column to the listbox in the “Changing an Order Record” dialog. You'll also add an edit box which automatically holds the grand total for this column. The edit box will appear below the listbox.

These controls will reference the data variables you just created.

Opening a Window

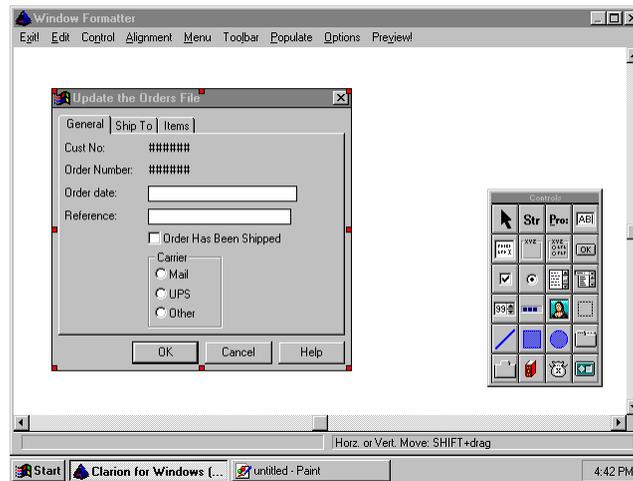
- ❑ Locate the Update Orders form in the Application Tree (it should be the 7th procedure from the top, when all the folders are expanded).
- ❑ Right-click on this procedure name, then choose Window from the popup menu. This allows you to edit the window contained within the procedure.



Editing a Window

When you first open the procedure, it should look something like the one below.

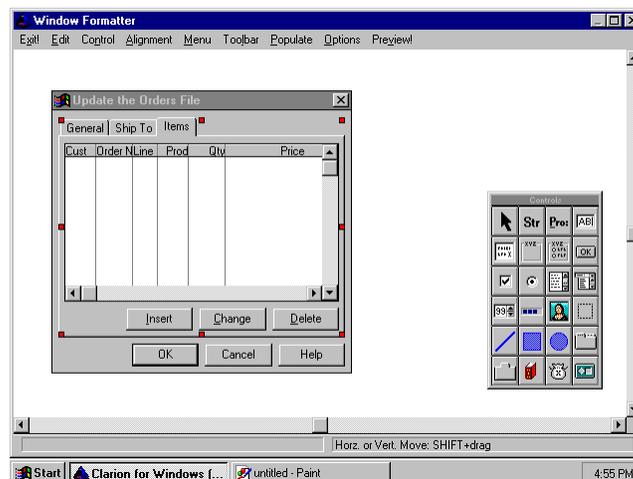
Notice the checkbox. This was provided by a pre-formatting option in the dictionary, for the Shipped field.



Select the Items Tab

This dialog is an update form for the Orders table. It includes a tab that displays related records in the Items table (the products that belong to this particular order, along with their quantity and price).

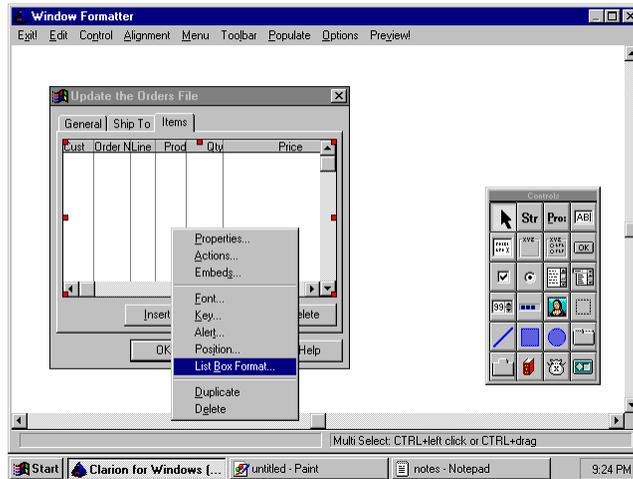
- Click on the Items tab, to bring it to the front.



Totaling a List Box -1-

First, you'll add a total for the price column.

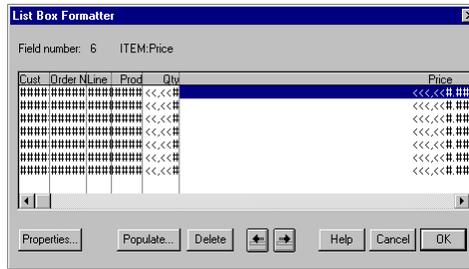
- Right-click the listbox, and choose List Box Format. This provides access to the List Box Formatter, which helps you populate and format fields and variables for the list box.



Totaling a List Box -2-

Place the line total field as the last field in the list box.

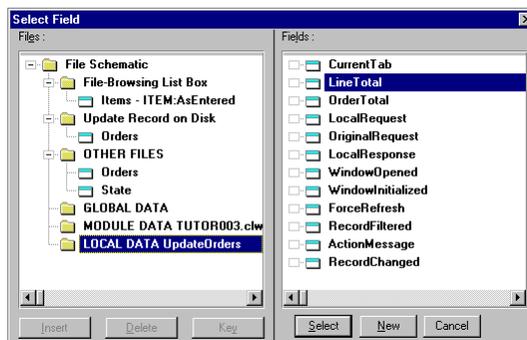
- Click on the first row of the last column (the Price column):



Totaling a List Box -3-

Choose the data variable you defined:

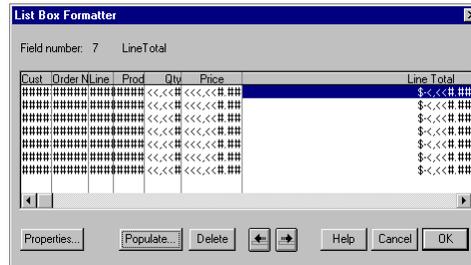
- Press the Populate button.
- Select Local Data.
- Select LineTotal.
- Press the Select button.



Totaling a List Box -2-

The listbox formatter also allows you to format the field. You can add resizable column lines, split a record to add a second “row,” set the listbox background color, and choose other options. For this exercise, just accept the field options set for the variable (currency formatting).

- Press OK to close the listbox formatter:



Editing the Window -1-

We need to resize a control in the window, to make room for another field.

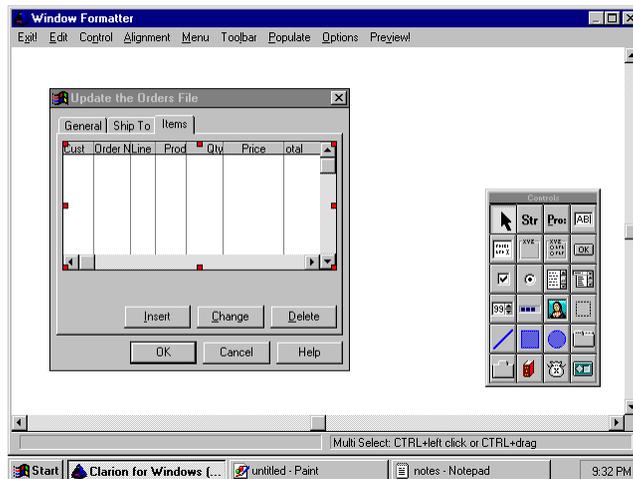
First, a word for developers who haven't spent a lot of time with visual design tools: you're going to spend a surprising amount of time “touching up” windows for size, alignment, and generally just “pretty-ing up.” This is just one of those “facts of life” related to windows programming, no matter what tool you use.

The great thing about Clarion for Windows is that the Application Wizard does so much more “real programming” that it frees you up for the “pretty-ing up” time.

Editing the Window -2-

You need to provide for a blank space below the list box, in which you'll place the Order Total field.

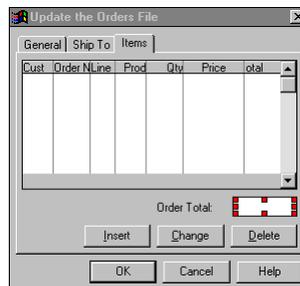
- Left click on the list box, and drag the center bottom handle up just a little bit—enough to fit an entry box.



Editing the Window -3-

Here's how you place a variable (or database field) in a window:

- Select the Dictionary Field tool from the tool box palette .
- Select Local Data.
- Select OrderTotal.
- Press Select.
- Click below the list box, in line with the left side of the Change (middle) button.



Editing the Window -4-

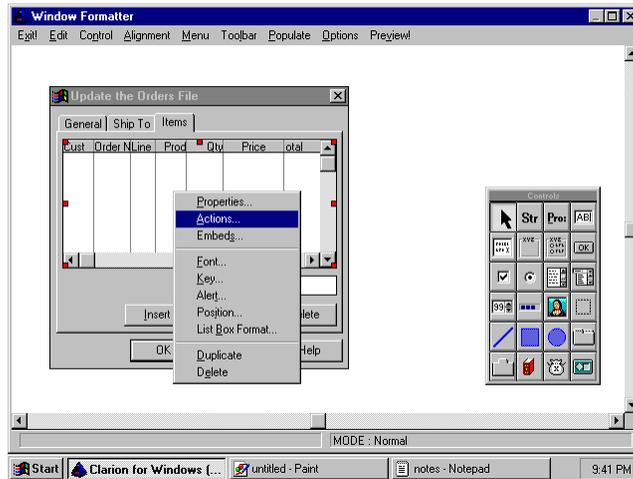
The Select dialog reappears, ready to place another field or variable.

- Press Cancel; you only need to place the one field.

The Template Interface -1-

Access the template interface for the browse list box control. It contains options which allow you to sum, count, or average any column in the listbox, and place the result in an edit box outside the list box.

- Right click the listbox.



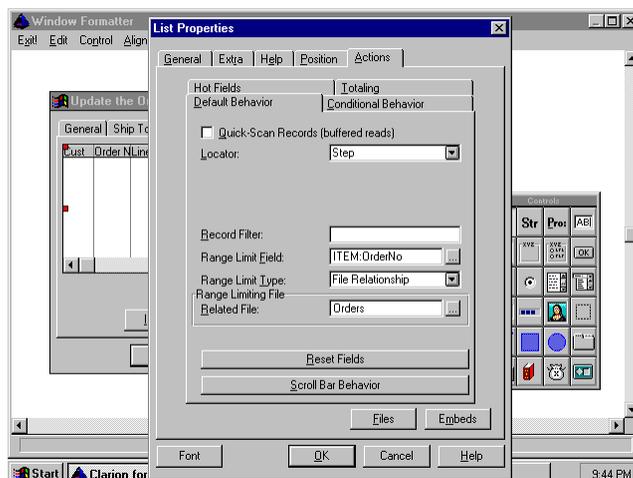
The Template Interface -2-

The Actions menu item displays the design-time user interface controls that the template writer included in the template to allow you to set the properties that direct the Application Generator to generate code to support the functionality you request. These controls appear on a property sheet which appears when you choose the menu item.

You can write your own templates. The Template Language Reference describes the various control structures, dictionary, and application symbols that you can access to intelligently generate code. It also describes the controls you can include in your own templates so that you or other members of your development team can choose exactly what code to generate.

The Template Interface -3-

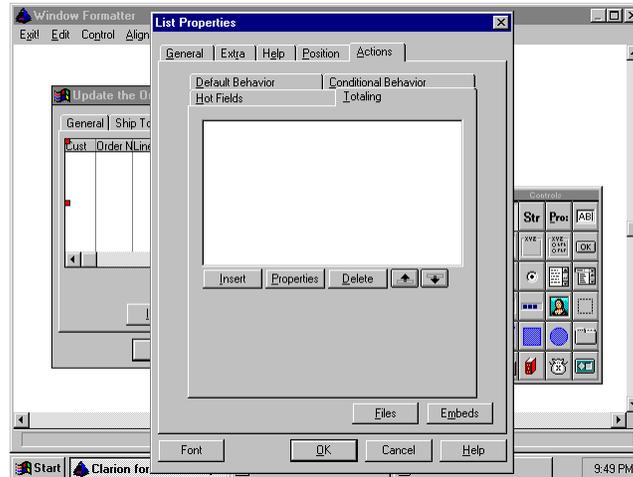
- ❑ Choose Actions.



The Template Interface -4-

To add a browse total, you need to access the controls on the Totalling sheet.

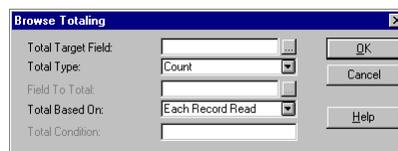
- Click the Totalling tab.



The Template Interface -5-

You can optionally total more than one column; to do so, you choose one at a time. For this exercise, you'll only choose one column to total:

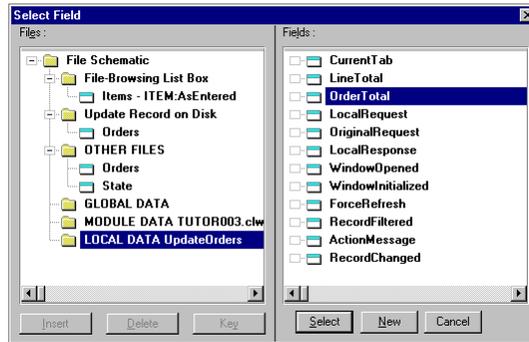
- Press Insert to open the Browse Totalling dialog. This dialog is part of the browse list box control template.



The Template Interface -6-

First you indicate the entry box to hold the total. You do this by selecting the variable that the entry box references, in this case, the Order Total field:

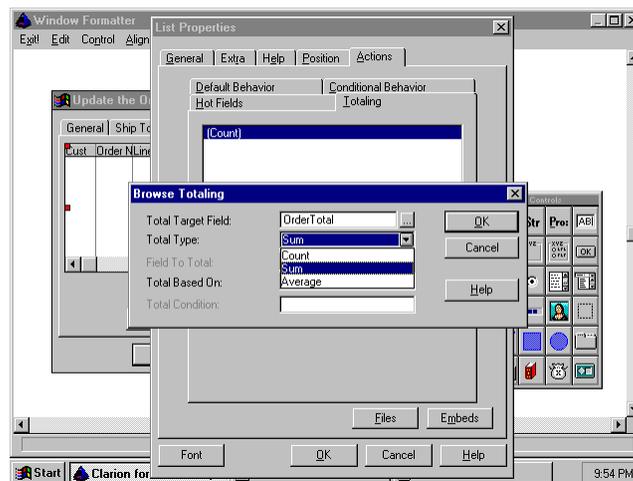
- Press the Ellipsis button next to Total Target field.
- Select Local Data from the Files list.
- Select OrderTotal from the Fields list.
- Press Select.



The Template Interface -7-

Next you indicate the type of totaling:

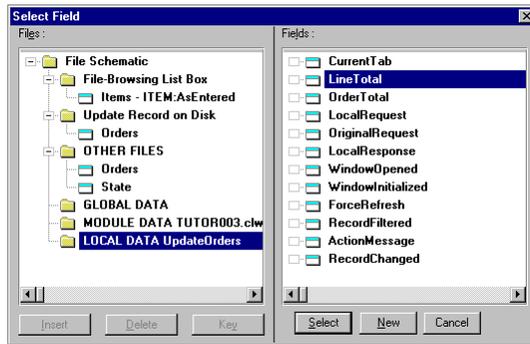
- Select Sum from the Total Type drop down.



The Template Interface -8-

Next, you indicate which column to total, referencing the variable the column contains, in this case, the Line Total:

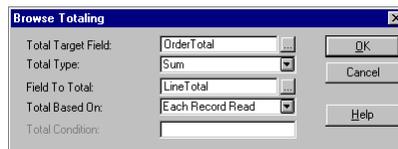
- Press the ellipsis button (...) next to Field To Total.
- Select Local Data.
- Select LineTotal.
- Press Select.



That's All Folks... -1-

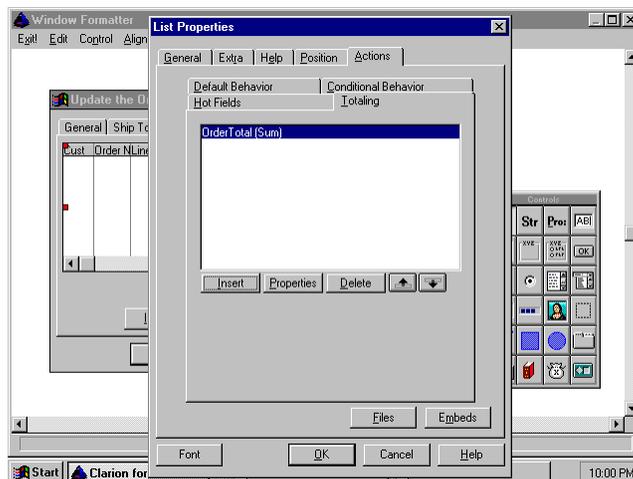
That's it. You're done. You're now an expert. You just have to close the dialogs and formatter windows, then recompile:

- Press OK to close the Browse Totaling dialog (this is a template dialog).



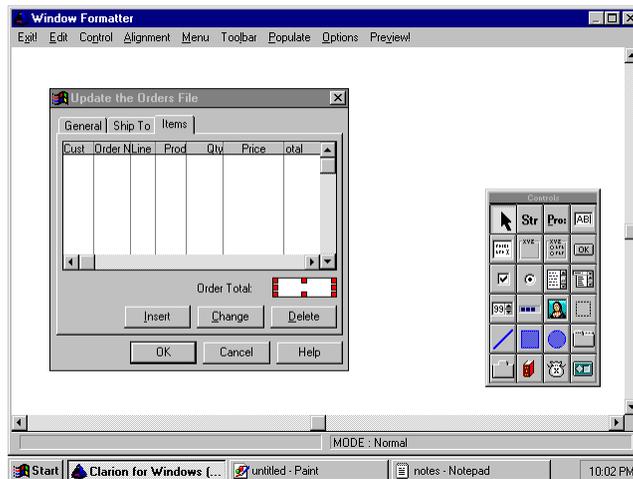
That's All Folks... -2-

- Press OK to close the List Properties dialog.



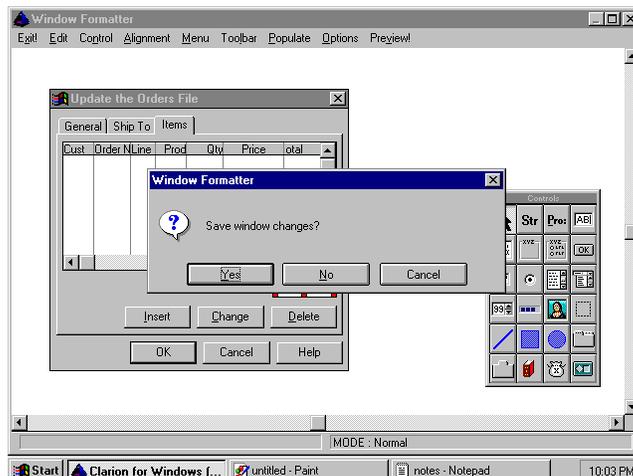
That's All Folks... -3-

- Choose Exit to close the Window Formatter:



That's All Folks... -4-

- Press Yes to save your changes to the Window structure:



That's All Folks... -5-

Save your file, then recompile and run the app!

- Choose File ► Save to save your work so far.
- Press the Make & Run button. This is an incremental compile, so only one module (the one containing the window) will be recompiled.

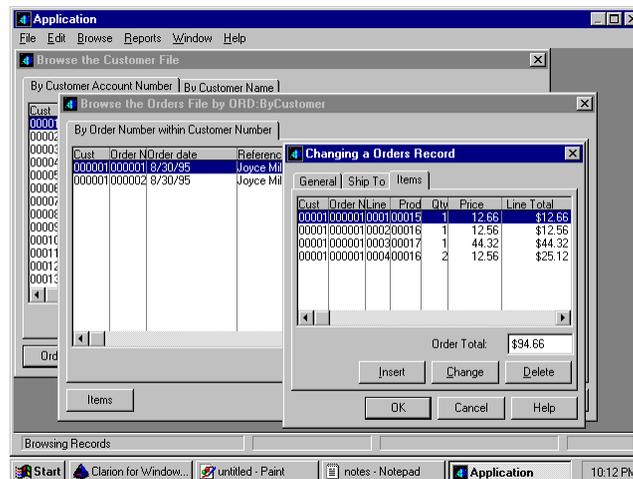
That's All Folks... -6-

Check your work. With the application running,

- Choose Browse ► Browse Customer Information File .
- Double-click on the first record in the list to display the Changing a Customer Record dialog.
- Click on the Orders tab to see the related Orders.
- Double-click on the first record in the list to display the Changing an Orders Record dialog.

That's All Folks... -7-

- Click on the Items tab to see the related Items. You'll find the Line Total and Order Total on this sheet.



Where to Go From Here -1-

Would you like to take a shot at creating an application from your own data files, now? Be sure to work with a copy of your database first, until you're more familiar with Clarion.

Do you have dBase or Clipper files? You can use the direct drivers. For Microsoft Access files, you can use an ODBC driver (note: you cannot use the Microsoft Office 4.x Access driver—it's designed **only** for use by Microsoft Office! Be sure your Access ODBC driver—ODBCJT16.DLL or ODBCJT32.DLL—is version 2.00.23.17 or higher. Select the driver in File Manager or Explorer and choose File/Properties).

Where to Go From Here -2-

If you'd like to continue doing exercises, we suggest reading the document containing the Getting Started manual. This is a 200 page book with a more comprehensive tutorial than the exercises you've followed here. You actually create the database, and build the application step by step.

There's no limit to the functionality you can include in a Clarion application. Be sure to look in your \CWEVAL\EXAMPLES subdirectories for several example apps. We included an icon in your Clarion program group for the **Tree** application, as a representative sample app.

Thank you for following the exercises.