

# ***CLARION 5***

## **Getting Started**

**COPYRIGHT 1994, 1995, 1996, 1997, 1998 by TopSpeed Corporation  
All rights reserved.**

This publication is protected by copyright and all rights are reserved by TopSpeed Corporation. It may not, in whole or part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from TopSpeed Corporation.

This publication supports Clarion 5. It is possible that it may contain technical or typographical errors. TopSpeed Corporation provides this publication “as is,” without warranty of any kind, either expressed or implied.

**TopSpeed Corporation**  
150 East Sample Road  
Pompano Beach, Florida 33064  
(954) 785-4555

**Trademark Acknowledgements:**

TopSpeed® is a registered trademark of TopSpeed Corporation.

Clarion 5™ is a trademark of TopSpeed Corporation.

Btrieve® is a registered trademark of Pervasive Software.

Microsoft® Windows® and Visual Basic® are registered trademarks of Microsoft Corporation.

Wise Installation System is a registered trademark of GLBS, Inc.

All other products and company names are trademarks of their respective owners.

# TABLE OF CONTENTS

<b>1 - INTRODUCTION</b>	<b>5</b>
<b>Welcome!</b>	<b>5</b>
User/Developer Levels .....	5
What You'll Find in this Book .....	7
Where to Find More Information .....	8
<b>Documentation Conventions</b>	<b>10</b>
Typeface Conventions: .....	10
Keyboard Conventions: .....	10
<b>Product Information</b>	<b>11</b>
Registering This Product .....	11
Technical Support .....	11
<b>The Setup Program</b>	<b>12</b>
System Requirements .....	12
Starting Setup .....	12
Setup Options .....	13
Starting Clarion .....	14
<b>2 - QUICK START TUTORIAL</b>	<b>15</b>
<b>Quick Start Wizard</b>	<b>16</b>
Creating a Dictionary and an Application .....	16
<b>Quick Load Wizard</b>	<b>21</b>
Adding a File .....	21
Adding a Relationship .....	23
<b>Procedure Wizards</b>	<b>26</b>
Using the Browse Wizard .....	28
<b>And Finally, the Application Wizard!</b>	<b>32</b>
Use the Dictionary Editor to set Wizard options .....	32
Using the Application Wizard .....	33
OK, What Did I Just Do? .....	35
What's Coming Next .....	36

<b>3 - CLARION PROGRAMMING CONCEPTS</b>	<b>37</b>
<b>The Tao of Clarion</b>	<b>37</b>
Clarion is a Programming Language! .....	37
Levels of Abstraction .....	37
Template Driven Programming .....	39
Clarion's Levels of Abstraction .....	40
The Clarion Key to Maximum Productivity .....	42
<b>Clarion's Development Environment</b>	<b>43</b>
Common Tools .....	44
Enterprise Tools .....	46
Where to next? .....	47

# 1 - INTRODUCTION

## Welcome!

Thank you for purchasing Clarion! You now hold in your hands the most powerful and flexible application development tool available, as you will soon see for yourself.

## User/Developer Levels

---

Clarion is an infinitely adaptable environment, written to offer fast solutions at every skill level: from the business owner to the enterprise development team. Whatever your level coming in, Clarion will help you take control of your company data—more cost-effectively, and up to three times faster than any other product out there today. Here's why . . .

### It's a wizard driven, automatic application developer

When you just need a “simple” application to maintain a database, you can literally do the job in minutes using Clarion. The key is the database dictionary. If the Application Generator knows what files or tables you want in the application, and how they're related, it can build an application. This is true regardless of where the files originated or in what format they are. So all you need to do is select one or more files, then indicate (when there are two or more files) the type of file relationships. The short **Quick Start** tutorial in this book demonstrates just how easy this whole process is.

The Application Wizard can then create a full featured application using the Clarion default application metaphor. We call this the *browse-form* metaphor, and it extends directly from the structure of your database. The application works like this: (1) The end user navigates the database—all or part of it, one data file or multiple related data files—by scrolling through a listbox, within which each item represents one record. The window in which this takes place is called a “browse.” (2) The end user selects a specific record in the list to perform an action, such as editing the data. This generally occurs in a separate window, in which the database fields appear in separate edit boxes. This is called an update “form.” A form may also accept new data. (3) Optionally, the end user can look up a value from a related table during form entry. This opens another browse in a separate window. The end user can select an item, closing the new window and placing the value in the edit box on the form, in one step. This is called a “lookup.”

That's the most general description. In addition, each browse window, navigable from the toolbar, opens on a separate thread with its own record buffer—which provides safer data handling. You can select among multiple key orders by **CLICKING** a tab. **DOUBLE-CLICKING** a record in the list opens an update form, with automatic concurrency checking (support for multi-user situations), and optional Referential Integrity constraint support (maintaining your database relationships).

*Anybody can do this. It just starts with picking a data file from a list.*

### **It's a visual development environment**

With Clarion, dropping a control in a window gives you a lot more than other Rapid Application Development tools, which typically let you add a user interface control, but then expect *you* to write the code to implement its associated functionality. With Clarion, you add a *template*, which contains the control, and all its required data elements and executable code. That means you don't have to write code—one **CLICK** places a complete business solution: a user interface control, and the code that enables it to do its job. Moreover, each template has its own user interface. When you view the properties for the template, you'll see an "Actions" tab. By checking a box, choosing a dropdown list item, or filling in an edit box, you can customize the behavior of the template so that it meets your needs *exactly*. You'll set "actions" for the templates at many places in the longer tutorial in this book. The **Application Generator** tutorial in the *Learning Clarion* book introduces you to all the Clarion RAD tools.

When you use the template interface to specify these behaviors, the Application Generator writes the code (Clarion language source code) that implements the behavior for you—and the code it writes for you is object-oriented, built from Clarion's *Application Builder Class (ABC) Library*, so the code is very compact and efficient. Using the templates, you can do an awful lot of custom programming without writing a single line of source code by hand.

The breadth and scope of the Clarion language can seem imposing at first. The Language Reference is about 900 pages. One of the great advantages of having the template system *plus* the ability to write code by hand is that you can *ease into* the language—as slowly as, as quickly as, as much as, or as little as you wish. If you don't like the way the templates solve a particular problem—you can use it at first, just to have something to do the job. Later, you can use the template interface, to modify it a little more to your liking. Finally, when you know the Clarion language, you can write a solution yourself, and have complete freedom to do it your way (Note: you can also buy third-party templates to solve problems yet another way).

*Power users, who may not normally write programs, can easily do this.*

## It's a complete programming language

At its most basic level, you can completely hand-code an application using Clarion's fourth generation programming language. Clarion has a high level of abstraction, so that it's very "readable," and a compact database grammar, so that you can easily handle a record or sets of records. You don't *have to* know Clarion to create an application... but if you do know how it works, it helps you understand what your applications are doing, and that helps you make better applications. The **Application Generator** tutorial in *Learning Clarion* has you write a small amount of your own Clarion code into the template-generated source, and the **Clarion Language** tutorial (also in the *Learning Clarion* book) introduces the Clarion language at a fully hand-coded level.

*Professional developers will really appreciate the Clarion language. It was designed from the ground up for business programmers. Yet for its relatively quick learning curve (as a high-level 4GL) you'll get blazing performance. The TopSpeed compiler technology turns all of your code—whether you wrote it or the Application Generator wrote it for you—into highly optimized machine code.*

**No matter which level you intend to work at, you're going to work a lot smarter if you first work through all the tutorials all the way to the end.**

## What You'll Find in this Book

---

The *Getting Started* book is your first introduction to the Clarion environment. The following lists the parts of this book and summarizes its content:

- |                                     |   |
|-------------------------------------|---|
| <b>Introduction</b>                 | <i>Chapter One:</i> introduces Clarion and provides instructions for installing Clarion on your computer.   |
| <b>Quick Start Tutorial</b>         | <i>Chapter Two:</i> a few quick steps with the Quick Start Wizard allow you create a complete application in <i>five minutes</i> . Invest <i>less than an hour</i> , and this chapter will show you how to use the Application Wizard and the Procedure Wizards to create a simple application, then extend its functionality into a sophisticated program that completely manages two related files. You'll accomplish all this <i>without writing a single line of code</i> . |
| <b>Clarion Programming Concepts</b> | <i>Chapter Three:</i> an overview of the Clarion concepts of application development. It describes all the tools of the development environment.  |

## Where to Find More Information

---

The first place to look for more information is the **How do I ... ?** section in the on-line Help. These topics answer many of the common questions that newcomers to Clarion have. Press the **How do I ... ?** button on the Help Contents page to get to this section.

The Help system in Clarion is very extensive. The hypertext help appears when you press the F1 key, or choose one of the commands on the **Help** menu. The on-line help is arranged by dialog box, to provide you with precise context-sensitive help when you need it.

There are also a number of books included with Clarion (printed and/or on CD in .PDF format):

### *Getting Started*

The book you're reading now.

### *Learning Clarion*

This book contains *two* step-by-step tutorials. The first is the Application Generator tutorial which provides a thorough introduction to all the tools in the Development Environment. The second introduces the Clarion programming language itself.

### *Application Handbook*

The handbook to the Clarion tools that you'll use the most during your application development. It describes the templates that ship with this product, and fully documents the *Application Builder Class (ABC) Library* used in the template-generated code.

### *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.

### *User's Guide*

Provides a task-oriented description of the development environment, arranged by its major components (on CD in .PDF format in the Professional Edition).

### *Programmer's Guide*

A collection of in-depth articles on various aspects of Clarion language programming and articles on customizing the development environment (on CD in .PDF format in the Professional Edition). It also 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. This book also provides in-depth information on all the file drivers available for Clarion.

#### *ReportWriter User's Guide*

Documents Clarion's ReportWriter for Windows™—a stand-alone end-user report writing tool (on CD in .PDF format in the Professional Edition—also available as a separate product).

#### *Enterprise Tools*

Documents the Data Dictionary Synchronizer, Version Control/Team Developer, and Data Modeller tools. This book also documents the Business Math Library (standard business and statistics functions) and Wise for Clarion (an end-user installation set creation tool). This book is only in the Enterprise Edition (some components are also available as separate products).

#### *Wizatron Handbook*

Documents the Wizatrons—Clarion's next-generation wizard technology which learn how you work and become your automated programming assistants. This book is only in the Enterprise Edition.

#### *Master Index*

A complete index listing for all printed and .PDF documentation with that edition of Clarion (printed only in the Enterprise Edition—on CD in .PDF format in all other editions).

All books on CD in .PDF format are accessible through the Adobe Acrobat Reader program, which you can also install from the Clarion CD. These .PDF format files are full-text searchable (and fully-indexed for fast searching) and can be printed on your printer if you wish.

**Important: if any part of the on-line help text conflicts with the printed documentation, the information in on-line help should take precedence.** TopSpeed Corporation makes every reasonable effort to ensure the printed documentation is up to date. However, the lead-time required by printers may create a lag in the documentation; while we can update the help files that ship concurrently with a product revision, printed materials must “catch up” later.

## Documentation Conventions

### Typeface Conventions:

---

<i>Italics</i>	Indicates what to type at the keyboard, such as <i>Enter This</i> .
SMALL CAPS	Indicates keystrokes to enter at the keyboard, such as ENTER or ESCAPE, or to CLICK the mouse.
<b>Boldface</b>	Indicates commands or options from a menu, or text in a dialog window. Note: this style can also utilize a different typeface to match the helvetica bold face which Windows uses as the system font.
LETTER GOTHIC	Used for diagrams, source code listings, to annotate examples, and for examples of the usage of source statements.

### Keyboard Conventions:

---

F1	Indicates a single keystroke. In this case, press and release the F1 key.
ALT+X	Indicates a combination of keystrokes. In this case, hold down the ALT key and press the X key, then release both keys.

## **Product Information**

### **Registering This Product**

---

Before you begin using Clarion, fill out and mail in the registration card that came in the package. This Business Reply Card makes you eligible to receive several important benefits. Once registered, you can use TopSpeed's Technical Support services, and you automatically receive new product announcements and update alerts.

### **Technical Support**

---

You can receive unlimited free technical support for Clarion on CompuServe Information Service and the Internet. TopSpeed employees, as well as TopSpeed Certified Support Partners (a group of volunteer Clarion users known as *Team TopSpeed*), will answer your questions in a timely manner. Additionally you will get advice and answers from other Clarion users. We strongly recommend that all our customers take advantage of these services.

- On CompuServe, GO TOPSPEED to access TopSpeed's support forum.
- On the Internet, access the *comp.lang.clarion* newsgroup through any USENET server.
- On the Internet, access the *topspeed.products* family of newsgroups which are hosted only on the *tsnews.clarion.com* news server.
- On the Internet, visit TopSpeed's Web page at <http://www.topspeed.com> for further Clarion resources.

Paid telephone technical support is also available from TopSpeed Corporation (954) 785-4556. Call Customer Service at (800) 354-5444 or (954) 785-4555 for more information on the various support programs TopSpeed offers.

## The Setup Program

The Setup program decompresses and copies the Clarion files to your hard drive.

- ◆ For all the target operating systems, it provides you with options for installing the various components, such as the example files.
- ◆ In all versions of Windows, it *asks* before updating the PATH statement in your AUTOEXEC.BAT file to include the Clarion directory.
- ◆ In Windows 3.x, it asks to install Program Manager icons for the Clarion development environment, Debugger, Help files, and ReadMe files.
- ◆ In Windows 95 and Windows NT, it asks to install all the Clarion development environment file icons to the Start Menu programs.

## System Requirements

---

You can run the Clarion development environment on any system that meets the minimum system requirements for Microsoft Windows 3.x, Windows 95™, or Windows NT. These are our recommended minimums:

- ◆ Windows 3.x, 8 Megabytes of RAM.
- ◆ Windows 95, 16 Megabytes of RAM.
- ◆ Windows NT, 32 Megabytes of RAM.
- ◆ Minimum of 12 to 35 Megabytes free hard disk space, depending on the Setup options you select.

The applications that you develop with Clarion will execute comfortably on computers that meet only the minimum requirements for these operating systems.

## Starting Setup

---

To start the Clarion Setup program in Windows 3.x:

1. Insert the Clarion installation CD into your CD-ROM drive.
2. From Program Manager, File Manager, or other shell program capable of launching a program, choose **File ► Run**.
3. Type `D:\SETUP` in the **Run** dialog (where *D:* is the drive letter assigned to your CD-ROM drive), and press the **OK** button.

The Setup program provides an introductory screen and other text information. It prompts you for your desired Setup Options.

To start the Clarion Setup program in Windows 95:

1. Insert the Clarion installation CD into your CD-ROM drive.
2. From the Start menu, choose **Settings ► Control Panel**.
3. Choose **Add/Remove Programs** then press the **Install** button.

The Windows 95 Wizard directs you through the installation process.

## Setup Options

---

After starting Setup, you'll see a set of wizard screens displaying a number of options.

1. Choose the Setup options offered by the wizard screens (such as the target drive and directory), pressing the **Next** button to move through the option screens.

Setup will install the main components of the Clarion Development Environment to a BIN subdirectory one level below the target directory you specify in the dialog.

The Clarion Setup program installs *all* its files to the target directory, and subdirectories beneath it. It installs *no* files to any other directory.

During the installation, progress bars will display as Setup copies the files.

2. Choose **Yes** or **No** when Setup asks whether to modify the PATH for you.

For Windows 3.x and Windows 95, the Clarion development environment requires that the BIN subdirectory be listed in the PATH environment variable. If you choose **No**, you *must* edit the AUTOEXEC.BAT file manually.

The only other change to any of your system files is that Clarion adds its own section to WIN.INI (Windows' initialization file) when you run it for the first time.

3. Choose **Yes** or **No** when Setup asks whether to install icons in a Clarion program group or on the Start menu.

4. Choose **Yes** or **No** when Setup asks whether to display the ReadMe file.

If you don't wish to read it right away, you'll find an icon for it in the Program Manager group (or the Start menu) which Setup creates for you. We recommend reading it as soon as Setup has copied all the files.

5. Press the **OK** button when Setup is done.

## Starting Clarion

---

To run Clarion, locate the Clarion icon in the Clarion program group and DOUBLE-CLICK on it, or on the Start menu, just CLICK on it:

The Clarion development environment appears, ready for you to begin work.

- ◆ You'll find a quick guide to the development environment parts—such as a diagram of the tool bar icons—in chapter three.
- ◆ Go on now to chapter two of this book to create *two* impressive applications—from data dictionary, to browse windows, to form windows—all in *less than an hour*.

# - QUICK START TUTORIAL

Clarion is a data-centric application development tool. Businesses run on data. That means Clarion is optimized to create business programs. Clarion's data-centric approach begins with the Data Dictionary.

In Clarion, you can create a data dictionary and a working application both at the same time—with no coding required. All you need to do is define the fields in your data file and the Quick Start Wizard creates a complete Windows application—in about five minutes if you're a fast typist! Your application has an update form procedure to maintain the file, a browse view on all the the sort keys and as many reports as there are keys.

Using the Quick Start Wizard, for each field in the data file you only have to define its name, display format picture, and key information—default values are provided for everything else. The Quick Start Wizard does everything else for you!

In this chapter:

- ◆ You'll use the **Quick Start Wizard** to create a Customer file and an application to maintain the file, then compile and run the program.
- ◆ You'll use **Quick Load**—another Clarion shortcut that lets you quickly add a file to a data dictionary.
- ◆ You'll define the relationship between two files in the **Dictionary Editor**, complete with Referential Integrity rules.
- ◆ You'll use the Application Generator to add more functionality to the application using one of Clarion's **Procedure Wizards**.
- ◆ Finally, you'll use Clarion's **Application Wizard** to create a second complete relational database application from the same data dictionary.

**This should all take about thirty minutes**—without *any* “coding” on your part. By the end of this chapter, you'll have two complete applications for a database containing two related files.

Welcome! Let's get started!

## Quick Start Wizard

### Creating a Dictionary and an Application

#### Starting Point:

You should have just opened the Clarion development environment. The Pick dialog should be open.

First, we'll create a C:\CLARION5\TUTORIAL directory for the applications you'll create from this book. This tutorial assumes that you installed Clarion in C:\CLARION5. If you used a different directory, you will have to modify the instructions accordingly.

#### Create a working directory

1. **Task switch** to the appropriate tool in your operating system (File Manager, Explorer, etc.) and **create** a new directory called *TUTORIAL* under the **CLARION5** subdirectory.
2. **Return** to Clarion.

#### Create your first Clarion application

1. On the Pick dialog, **press** the **New...** button.

The **New** dialog appears. It's a standard Windows *Open File* dialog, allowing you to change the directory and type in the filename.



2. **Select** *Application (\*.app)* from the **Save as type** dropdown list.
3. **Select** the C:\CLARION5\TUTORIAL directory.
4. **Type** *QWKTUTOR* in the **File name** field.

Quick Start Wizard uses this as both the application and dictionary name to create *QWKTUTOR.DCT* (the data dictionary file) and *QWKTUTOR.APP* (the application file). You *can* use long filenames, but this tutorial won't—for the benefit of any Windows 3.1 users.

5. Check the Use Quick Start box, then press the Save button.

The Clarion Quick Start dialog appears.



### Define the data file

1. Press TAB twice, type *Customer* in the Data File Name field and press TAB again.

Quick Start Wizard uses this name as the data file name. Notice that it takes the first three letters, *CUS*, and places them in the Prefix field. A file's Prefix makes field names unique across multiple files which share common field names.

2. Press TAB to accept *CUS* as the Prefix.

Next, you choose a File Driver. This field defaults to TOPSPEED (one of Clarion's two proprietary file systems).

3. Press TAB to accept TOPSPEED as the File Driver.

This takes you to the list box where you define the fields.

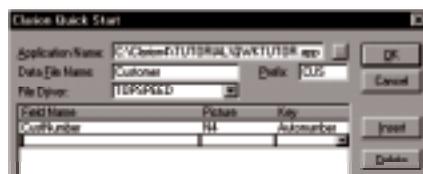
**NOTE:** Under Windows 3.1, you must have SHARE.EXE or the Windows VSHARE.386 driver loaded to use the TopSpeed file driver. Newer versions of Windows handle this for you.

4. Type *CustNumber* in the first row of the Field column, and press TAB.

This creates a field named *CustNumber*. Quick Start Wizard also uses this name for the default Prompt and Column Heading. The Prompt is used whenever you place the field on a window. Column Headings are used in reports and list boxes.

5. Type *N4* in the Picture column, and Press TAB.

This specifies default display picture for window and report controls, which implicitly declares the data type for the Quick Start Wizard.



6. In the Key column, press the DOWN-ARROW to display the choices. Highlight *Autonumber*, then press TAB.

This specifies that a unique key will be created with this field as its component. By specifying *Autonumber*, your application will ensure that each customer has a distinct number, which will automatically increment every time a new record is added. Quick Start Wizard creates browses and reports based on every key created. This key will scroll through records sorted by *CustNumber*.

The cursor is automatically positioned on the next row, allowing you to define the next field.

**7. Type *Company* in the **Field** column, and **press** TAB.**

This creates a field named *Company* to use as the Customer's company name.

**8. Type *S20* in the **Picture** column, and **press** TAB.**

This specifies the default display pictures for window and report controls. In this case, the data type is a twenty-character STRING field.

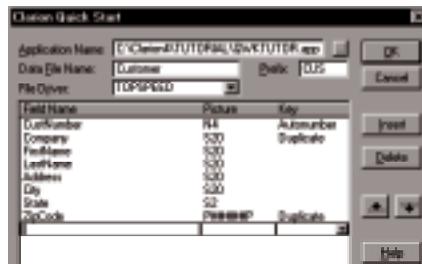
**9. In the **Key** column, **press** the DOWN-ARROW to display the choices. **Highlight *Duplicate***, then **press** TAB.**

This specifies a key will be created that allows duplicate entries. This allows you to have two Customers with the same company name.

**10. Create** the remaining fields in the same manner, following this table:

FIELD	PICTURE	KEY
FirstName	S20	(no key)
LastName	S20	(no key)
Address	S20	(no key)
City	S20	(no key)
State	S2	(no key)
ZipCode	P#####P	Duplicate

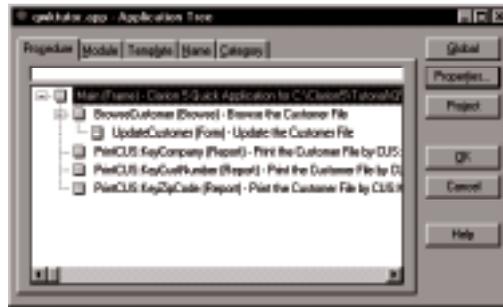
Notice the picture (P#####P) specified for the ZipCode field. It is a pattern picture that formats the digits for a five-digit U.S. zip code.



## Finish Quick Start Wizard and create the application

1. When you have defined all the fields, **press** the **OK** button.

Quick Start Wizard first asks if you are done defining fields. When you press the **OK** button, it then creates your application and displays the **Application Tree** dialog.



Notice the structure of the application. At the top of the tree is the *Main* (Frame) procedure, which creates an MDI (Multiple Document Interface) application frame. This procedure contains the menu which calls the other procedures.

Below the *Main* Procedure there is a *BrowseCustomer* (Browse) procedure with an *UpdateCustomer* (Form) procedure attached. The browse displays the data based on all three keys you specified when defining the fields. You also see three reports based on those keys.

The application is complete, without requiring any more information from you.

The toolbar icons in order from left to right are: Pick, New, Open, Save, Make, Run, and Debug



The Run Button

2. Choose **Project** ► **Run** (or **press** the *Run* button on the toolbar).

The Application Generator creates the source code for your application, then it compiles, links, and executes the resulting program.

*Congratulations!* Your application is running.

## Explore the application

First you need to add some records.

1. Choose **Browse** ► **Browse the Customer file**.
2. **Press** the **Insert** button on the toolbar.

The runtime toolbar icons from left to right are: Top of File, Page Up, Up, Locate, Down, Page Down, End of File, Select, Insert, Change, Delete, Ditto, and Help.

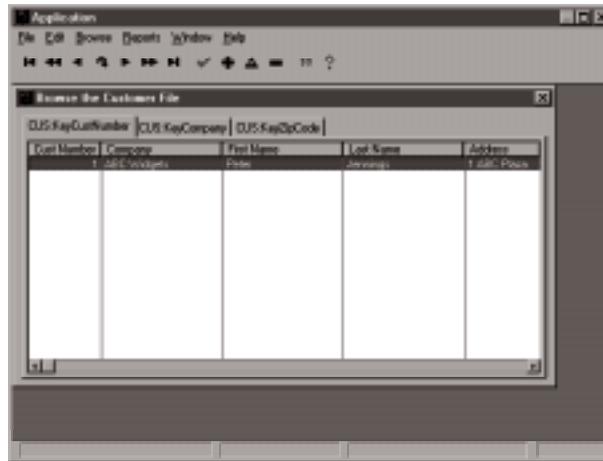


3. **Type** in the first customer's data then **press** the **OK** button to close the form window and save the record to the data file.

This demonstrates Clarion's default *Browse-Form* application metaphor. The Browse list displays the records in the data file, and the Form procedure is used to update (add, change, or delete) the individual records in the file. Clarion can also support the *Scrolling Form* metaphor where the data records are directly edited in the browse list (we'll get to this later).

4. **RIGHT-CLICK** on the browse list and **choose Insert** from the popup menu and add several more records.

Notice that each time you add a new Customer file record, the CustNumber field value automatically increments by one from the largest number added previously. Look at the different sort orders in the browse procedure, then resize the display columns and use the horizontal scroll bar to examine data in the fields that do not appear in the list initially. You can even print the reports, if you want.



5. **Choose File ► Exit** to exit your application.

You can also try running your program (QWKTUTOR.EXE) from the File Manager or Explorer—it's a true Windows executable that does not require the Clarion environment to run.

Now we will save your work and exit the Application Generator to modify the data dictionary we just created. Next, we will add a second, related file to the data dictionary.

6. **Choose File ► Close**, or **press** the **OK** button to close the Application Generator (make sure you save your work when asked).

## Quick Load Wizard

You can add a file to the data dictionary very quickly using the Quick Load option in the Dictionary Editor. Quick Load functions in a similar manner to the Quick Start Wizard—define the field (names, pictures, and keys), and a file definition is created in the dictionary. This option is available to you every time you add a new file to a dictionary.

### Adding a File

In this section, we will add a file to store the phone numbers for customers. This will allow us to have many phone numbers for each customer (creating a one-to-many relationship).

#### Use Quick Load in the Dictionary Editor

1. Choose **File ► Open** (or press the *Open* button on the toolbar).

The **Open** dialog appears.

2. Select the *Dictionary (\*.dct)* from the **Files of type** dropdown list, select the *QWKTUTOR.DCT* file, then press the **Open** button.



The **Dictionary Editor** dialog appears.

3. Press the **Add File** button.

A dialog appears asking, “Do you want to use Quick Load?”



4. Press the **Yes** button.

The **Clarion Quick Load** dialog appears. Notice that it is very similar to the Quick Start Wizard dialog.



5. Select the **Data File Name** field, type *Phones* in it, then **press** **TAB**.
6. **Press** **TAB** to accept *PHO* as the **Prefix**.

Next, you are prompted to choose a File Driver.

7. **Press** **TAB** to accept **TOPSPEED** as the **File Driver**.

This takes you to the list box where you define the fields.

8. **Type** *CustNumber* in the first row of the **Field Name** column, and **press** **TAB**.

This creates a field named *CustNumber*. This field is the link to the *CustNumber* field in the *Customer* file. Using the same field names makes it easier to link the two files in a relationship. In the generated Clarion code, adding the file's prefix to field labels (separated by a colon) creates unique names for fields with the same name in separate files. Therefore, this field will actually be called *PHO:CustNumber*, while the field in the *Customer* file is called *CUS:CustNumber*.

9. **Type** *N4* in the **Picture** column, and **press** **TAB**.
10. In the **Key** column, **press** the **DOWN-ARROW** to display the choices, **highlight** *Duplicate*, then **press** **TAB**.

This specifies a key that allows duplicate entries, enabling you to have more than one record for each customer. This allows us to create a One to Many relationship between the two files (*one* Customer may have *many* Phones).

The cursor appears on the next row, allowing you to define the next field.

11. In the **Field Name** column on the next row, **type** *Area*, then **press** **TAB**.
12. In the **Picture** column, **type** *P(###)P*, then **press** **TAB** twice (don't define a key) to move on to define the next field.

This formats the *Area* field as a three digit number surrounded by parentheses (the standard notation for telephone area codes in North America).

13. **Create** the rest of the fields, following this table:

FIELD	PICTURE	KEY
Phone	P###-####P	(no key)
Description	S20	(no key)

14. When you have defined all the fields, **press** the **OK** button.

Quick Load pops up a dialog telling you it will create the file and offering you the opportunity to go back and define more fields.

15. **Press** the **OK** button.

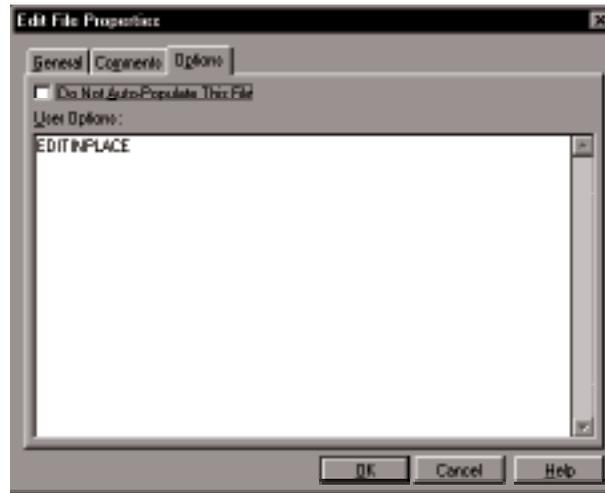
Quick Load now creates your file definition and adds it to the dictionary.

### **Set the new file to use the Scrolling Form metaphor**

Since there are only four fields, this file is a good example of the type of file for which editing the data right in the scrolling list is most appropriate, instead of using the Browse-Form metaphor.

1. **RIGHT-CLICK** on *Phones* in the **Files** listbox and **select Properties** from the popup menu.
2. **Select** the **Options** tab and in the **User Options** text box control **type** in *EDITINPLACE*.

This tells Clarion's Wizards to generate code to allow you to directly edit the file's data as it appears in the Browse list box.



4. **Press** the **OK** button.

## **Adding a Relationship**

---

Obviously, we want the Phones file to contain the phone numbers of the Customers. This means there must be a relationship between the two files. In this case, one Customer can have many Phones, making this a "One to Many" relationship. To define this relationship, we must link the files together in the data dictionary to provide the Application Generator with the information necessary to access the related records.

## Set the Relationship for the two files

1. **Highlight** *Customer* in the **Files** listbox, then **press** the **Add Relation** button.

The **New Relationship Properties** dialog appears. This is where you define the relationship.



2. Make sure the **Type** field is set to 1:MANY.
3. In the **Primary Key** field, **press** the DOWN-ARROW key to display the choices, **highlight** *KeyCustNumber*, then **press** TAB.

This is the key on the Customer file (the One side of the relationship) that will be used to link the two files.

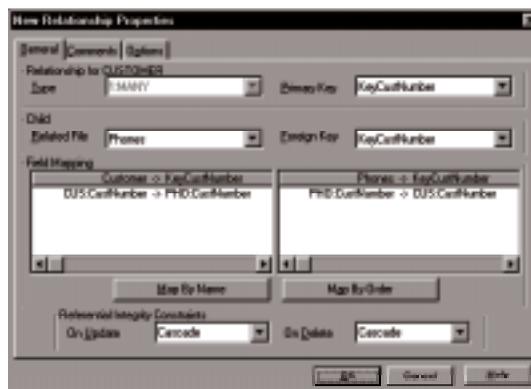
4. In the **Related File** field, **press** the DOWN-ARROW to display the choices, **highlight** *Phones*, then **press** TAB.
5. In the **Foreign Key** field, **press** the DOWN-ARROW to display the choices, **highlight** *KeyCustNumber*, then **press** TAB.

This is the key on the Phones file (the Many side of the relationship) that will be used to link the two files.

Next, the linking fields in the keys must be “mapped” so the Application Generator can know exactly which fields in the two files are related to each other. Since we used identical field names, this is easy.

6. **Press** the **Map By Name** button.

The linking fields between the two files appear in the two **Field Mapping** list boxes. This is one reason to name linking fields the same.



## Set Up the Referential Integrity Constraints

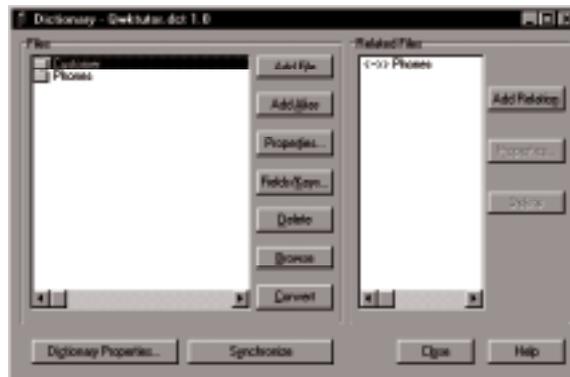
1. In the **Referential Integrity Constraints** group box, **choose Cascade** from the **On Update** dropdown list.
2. In the **Referential Integrity Constraints** group box, **choose Cascade** from the **On Delete** dropdown list.

The generated source code will automatically maintain referential integrity between the files.

The tutorial in the *Learning Clarion* manual, the *Using the Dictionary Editor* chapter in the *User's Guide*, and the *Database Design* article in the *Programmer's Guide* all have discussions of relational database theory which will explain the concept of Referential Integrity in more detail.

3. **Press OK** to close the **New Relationship Properties** dialog.

The Dictionary now appears like this:



4. **Choose File ► Save**, or **press the Save** button on the toolbar to save the data dictionary.
5. **Choose File ► Close**, or **press the Close** button in the **Dictionary** dialog to close the dictionary.

## Procedure Wizards

In addition to the Quick Start Wizard, Clarion also has Procedure Wizards that create Browse, Form, or Report procedures for you just as easily as the Quick Start Wizard creates an entire data dictionary and application. These procedure-level Wizards are available to you whenever you create a new procedure in an existing application.

Now we will use the Browse Wizard to create a browse procedure for the Phones file. It could also create an update procedure at the same time, but it won't this time because we told it to use the scrolling form metaphor (edit-in-place) for this file.

### Load the Application Generator

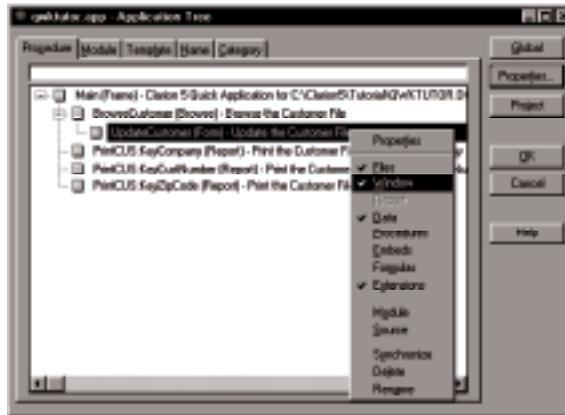
1. **Choose File** ► **Pick** (or **press** the *Pick* button on the toolbar).
2. **Select** the **Application** tab, **highlight** `c:\clarion5\tutorial\qwktutor.app`, then **press** the **Select** button.

The **Application Tree** dialog appears.

### Edit the Form Procedure

1. **Highlight** *UpdateCustomer (Form)* in the Application Tree then **RIGHT-CLICK** to display the popup menu.

This menu allows you direct access to the Application Generator's tools without requiring you to open the procedure's Properties dialog first.



The popup menu displays a check mark beside tools that contain data.

2. **CLICK** on **Window**.

The Window Formatter appears. Here you can visually edit the window and its controls.

3. **Choose Control** ► **Push Button**, or **CLICK** on the Button tool in the Controls toolbox (the one with the icon that looks like an **OK** button).

4. **CLICK** near the bottom left corner of the window to place the new Button control below the tab sheet.



5. With the new button selected, **RIGHT-CLICK** to display the popup menu then **choose Properties** to call the **Button Properties** dialog.

6. **Type *Phones*** in the **Text** field.

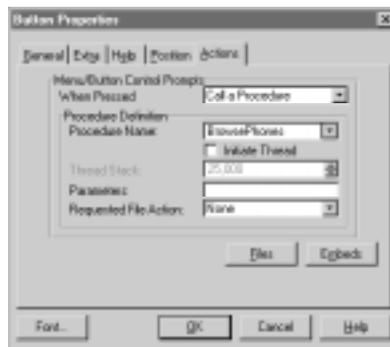
This changes the text that will appear on the button face.

7. **Select** the **Actions** tab.

This tab is where you specify what the control does. In this case, we want it to call a Browse procedure to display all the Phones file records that are related to the currently displayed Customer file record.

8. **Choose Call a Procedure** from the **When Pressed** dropdown list.

The Procedure Definition group box appears on the Actions tab.



9. **Type *BrowsePhones*** in the **Procedure Name** field then **press** the **OK** button.

This names the procedure to call when the user presses the button.

10. **Choose Exit!** from the menu to return to the Application Tree and **press** the **Yes** button when asked to save the window changes.

The BrowsePhones procedure appears in the tree as a (ToDo) item.

## Using the Browse Wizard

Procedure Wizards generate entire procedures based on minimal information that you provide in response to a series of Wizard dialogs that ask for one piece of information at a time.

### Call the Browse Wizard

1. **DOUBLE-CLICK** on the *BrowsePhones (ToDo)* procedure.

The **Select Procedure Type** window appears. This allows you to choose the procedure template that the Application Generator uses to create source code.

2. **Highlight** the **Browse** procedure Template and check the **Use Procedure Wizard** box (in the bottom-left corner of the dialog).

Checking the **Use Procedure Wizard** box is the key to using the procedure Wizards.



3. **Press** the **Select** button.
4. After you read the first Wizard dialog's information, **press** the **Next** button to begin creating the procedure.

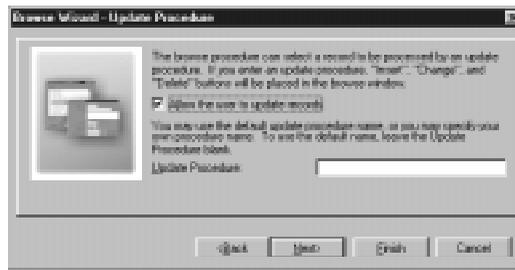


5. **Type** *Phones* in the entry control (or **press** the ... button and **select** it from the list) in response to the question “Which file do you want to browse?” then **press** the **Next** button.



6. **Clear** the **Browse using all record keys** check box, **press** the ... button and **select** *PHO:KeyCustNumber* from the list, then **press** the **Next** button.

This specifies the key used to sort the records for display in the list box.



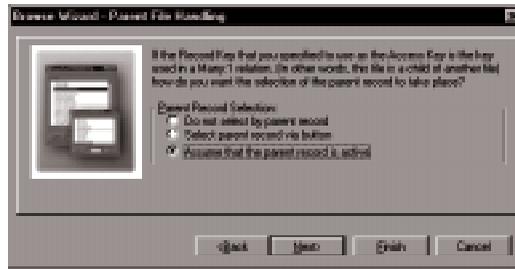
This next Wizard dialog allows you to name the procedure that the Browse Wizard will create to update the Phones file records. This is unnecessary in this case, since the Wizard will create code to edit this file in-place, eliminating the need for an Update Procedure. However, if we had not specified EDITINPLACE in the file's User Options, the Browse Wizard would actually call the Form Wizard to create the update Form procedure for us. By naming an Update Procedure here, you can save yourself the “work” of calling the Form Wizard yourself.

7. Leaving all defaults set, just **press** the **Next** button.
8. Leave the **Call update using popup menu** box checked and **press** the **Next** button.

This gives users the option of using a popup menu (that appears when they RIGHT-CLICK on the list box) to call the update procedure (or the edit in place functionality).



9. **Clear** the **Provide buttons for child files** check box then **press** the **Next** button.  
The *Phones* file has no child files—it is the child of the *Customers* file.



10. **Select** the **Assume that the parent record is active** radio button then **press** the **Next** button.

This ensures that the procedure is set up to only display the *Phones* file records that are related to the *Customer* file record currently in memory at the time the procedure is called.



11. **Clear** the **Provide a "Select" button** check box then **press** the **Next** button.

Since this procedure will not be used to lookup data for an entry field, the *Select* button is unnecessary.

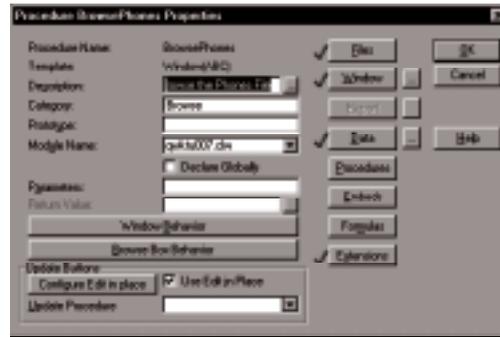


12. Leave the **Which Control Model should the Application Use** radio button set to *Toolbar* and **press** the **Next** button.

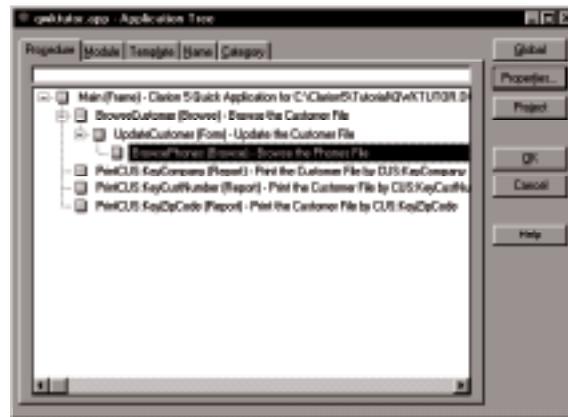
This gives the *BrowsePhones* procedure a consistent look and feel with the *BrowseCustomer* procedure. It will only use the *Toolbar* buttons and pop-up menus for *Insert*, *Change*, and *Delete* actions.

13. **Press** the **Finish** button.

The Browse Wizard now creates the Browse procedure for the Phones file. If we had not specified EDITINPLACE in the Phones file's User Options, the Browse Wizard would also have created an associated update (Form) procedure to maintain the Phones file records. When finished, it leaves you on the new Browse procedure's **Procedure Properties** window.



14. Press the **Ok** button to return to the Application Tree.



Notice the new procedure on the Application Tree that was created for you.

15. Choose **Project > Run** (or press the *Run* button on the toolbar).

Now you have a complete relational database application, featuring multiple browse orders on the parent file, browsing on the child file limited to the range of records related to one parent record, and Referential Integrity code to ensure that your database cannot be corrupted with “orphan” child records. Push the new Phones button on the UpdateCustomer Form and add records to the Phones file for each of your Customers.

16. When you have finished examining your new application, close it and **return** to the Clarion development environment.

17. Choose **File > Close** or press the **OK** button to close the Application.

## And Finally, the Application Wizard!

Quick Start Wizard and the Browse, Form, and Report Wizards are all powerful tools. However, Clarion has another, even more powerful Wizard in its arsenal of super-productivity tools.

The Application Wizard builds a complete application from an existing data dictionary. The data dictionary it uses may contain as many files as you like. The files should be fully defined, complete with all the file relationships, Referential Integrity, and Data Integrity rules that the Dictionary Editor allows you to define.

To facilitate using the Application Wizard, the Dictionary Editor has property options on each file, field, and key that are expressly designed to interact with the Application Wizard (and all the procedure Wizards) to allow you to customize the application it creates for you.

### Use the Dictionary Editor to set Wizard options

---

1. **Choose File** ► **Pick** (or **press** the *Pick* button on the toolbar).
2. **Select** the **Dictionary** tab, **highlight** `c:\clarion5\tutorial\qwktutor.dct`, then **press** the **Select** button.

The **Dictionary Editor** dialog appears.

3. **Highlight** the *Customers* file then **press** the **Fields/Keys...** button.
4. **Select** the **Keys** tab, **CLICK** on *KeyCustNumber* to **highlight** it, then **press** the **Properties** button.

The **Edit Key Properties** dialog appears.

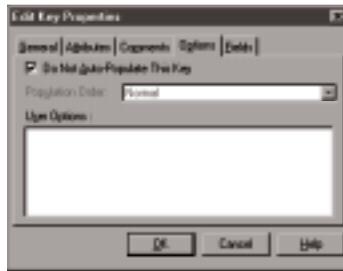
5. **Select** the **Options** tab.

The settings on this tab are all used to specify the behavior of the Wizards. The options in the upper half of the window are used with the standard set of Templates that Clarion ships. The **User Options** entry box is for the use of third-party add-on templates, so they may allow you to specify any constraints or options for their Wizards that are not covered by the standard options.

6. **Check** the **Do Not Auto-Populate This Key** box.

Checking this box tells the Application Wizard not to use this sort key when creating a Browse procedure for the file. This will eliminate the *CUS:KeyCustNumber* tab in the Customer file's Browse procedure.

There is a lot more than this that you can do in the Dictionary Editor to customize the output from all the Wizards. After you finish the tutorial, look at the *Optimizing the Wizards* section of the *Application Handbook* for more information on this important subject.



7. Press the **OK** button.
8. In the **Field / Key Definition** dialog, press the **Close** Button.
9. Press the **Close** Button in the **Dictionary** dialog then, when prompted, press the **Yes** button to save the dictionary file.

That's all it takes. Next, we will let the Application Wizard generate the entire application.

## Using the Application Wizard

---

The Application Wizard generates entire applications based on the minimal information that you provide in response to a series of Wizard dialogs that ask for one piece of information at a time.

### Create a New Application

1. Choose **File** ► **New** ► **Application**.

The **New** dialog appears.

2. Select the *C:\CLARION5\TUTORIAL* directory.
3. Type *APPWIZ* in the **File name** field and **clear** the **Use Quick Start** box.
4. Press the **Save** button.

The **Application Properties** dialog appears.



5. Press the ... button to the right of the **Dictionary File** entry box.

The **Select Dictionary** dialog appears.

6. **Highlight** the *c:\clarion5\tutorial\qwktdict.dct* file and press the **Open** button.

8. Check the Application Wizard box, then press the OK button.



9. Press the Next button.



10. Press the Next button.

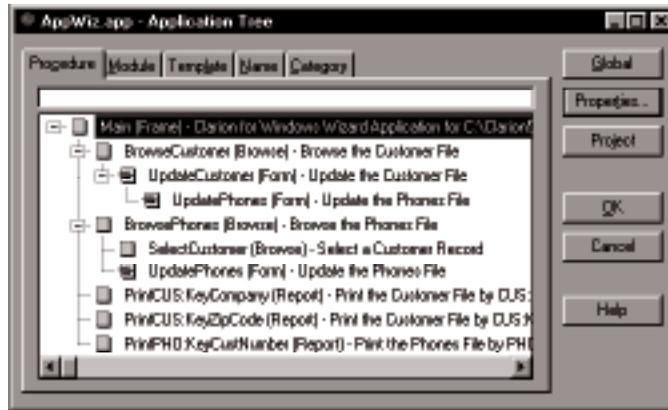


11. Press the Next button.



12. Press the Finish button.

The Application Wizard creates the application.



**13. Choose Project ► Run** (or **press** the *Run* button on the toolbar).

*Congratulations!* Your second complete application is running.

Now you can explore this new application and compare it to the application you already created. You will notice that there are some minor differences between the two. These differences are there simply because we built the first application one piece at a time instead of using the Application Wizard.

## OK, What Did I Just Do?

Let's do a quick recap now of what you've accomplished:

- ◆ You used the Quick Start Wizard to define a Customer file which the Quick Start Wizard used to create a Clarion Data Dictionary and Application.

When you compiled and ran that application, you found that the Quick Start Wizard had created a full-featured multi-threaded MDI application, complete with browse lists and update forms to maintain the data file, and even a set of simple reports—all without requiring you to write a single line of code!

- ◆ You used the Quick Load feature in the Data Dictionary to very quickly add another file definition.
- ◆ You specified that the new file would use the Scrolling Form metaphor (edit-in-place) for update instead of the default Browse-Form metaphor.
- ◆ Then you defined the relationship between the two files and specified the Referential Integrity rules governing the relationship.
- ◆ Next, you used the Browse Procedure Wizard to create an updatable (edit-in-place) Browse list for the new file.

When you compiled and ran the updated application, you found that the Browse Procedure Wizard had created an updatable (edit-in-place) Browse list for the new file which you called from the single button control you placed on the Customer file's update dialog—and you still haven't had to write a single line of code!

- ◆ Finally, you used the Application Wizard to create a second complete relational database application from the same data dictionary.

When you compiled and ran the new application, you found that the Application Wizard had created had created another full-featured multi-threaded MDI application, complete with browse lists and update forms to maintain both data files—and it did it all without requiring you to write a single line of code!

In about thirty minutes—without *any* “coding” on your part—you've created two complete applications for a database containing two related files. That's a real accomplishment!

*If you can do this much with Clarion's “shortcuts,” what can you expect to accomplish with all the other tools Clarion provides?*

## What's Coming Next

---

Besides the Wizards, Clarion also gives you a host of other productivity tools in the Application Generator, starting with the Control, Extension, and Code Templates that can add even more functionality to any procedure. These tools make modifying and customizing procedures just as easy as using the Wizards—and *none* of them require that you write any code!

The very next chapter of this book discusses **the underlying application development philosophy** which Clarion expresses. Please read it carefully. This is important background material because, as you have just seen, Clarion is a different kind of development tool than others. *Understanding the Clarion philosophy will make the learning curve easier to ascend.*

The **Application Generator Tutorial** is in the *Learning Clarion* book. This “hands-on” tutorial walks you through creating an Order Entry system “from scratch” *without using the Wizards*. This demonstrates all the Clarion tools, and will also show you how much power you have when you finally do write some source code yourself to provide some “non-standard” functionality.

Continue on! You've only just skimmed the surface, *and there's a lot more!*

# 3 - CLARION PROGRAMMING CONCEPTS

## The Tao of Clarion

The Tao (pronounced “dow”) means “the Way” in Chinese. Clarion is an entirely **new way** to create Windows applications—as the applications that you just created in about 30 minutes demonstrate! However, Clarion is much more than just its Wizards, so now we’ll take some time to explain the core concepts behind the toolset that is Clarion.

## Clarion is a Programming Language!

---

At the foundation of Clarion (the development tool) is the Clarion programming language—a fourth-generation, general-purpose (fully object-oriented) programming language which has been highly optimized since its inception for business (database) application development. The Clarion language is the real tool that you now hold in your hands. But Clarion is much more than just the Clarion language—it is also a complete toolset designed specifically for business database application creation and maintenance.

On top of the Clarion programming language is the Clarion development environment—a completely integrated set of tools that are specially designed for *Rapid Application Development* (RAD), *Rapid Application Maintenance* (RAM), and *Rapid Application Enhancement* (RAE). These tools are all geared towards generating Clarion language source code for you, so you don’t have to write it yourself. That’s what makes Clarion so powerful—although you *can* write your own Clarion language code, the toolset can generate (and re-generate) most “standard” code *for* you, leaving you free to concentrate on your application’s design issues.

## Levels of Abstraction

---

There is a principle that, the further you can get away from the “bare metal” of the computer (pure binary), the more productive you can be.

### A Bit of History

The very first digital computers were programmed in binary, one machine language instruction at a time, and the programmer’s job was very difficult. Then Assembly language came along to make the programmer’s job easier by making the program code readable by a human being. Each Assembly language statement generated exactly one machine instruction.

Early Assembly programmers noticed that they were consistently writing the same sequences of instructions to perform standard tasks. Therefore, they created Third Generation languages (3GL—like Fortran, C, PL/1, etc.) so that each 3GL statement would generate multiple machine instructions for these standard tasks and make programmers more productive. By doing this, they started operating at a higher level of abstraction away from the “bare metal” of the computer. The drawback was that only “standard” sequences of machine code were generated, so some of the total flexibility that’s possible when you work directly in Assembly language was lost.

### Present Day—4GL

This same basic process has occurred to bring about the advent of Fourth Generation languages (4GL) like Clarion. For example, the Clarion language automatically handles all the standard Windows chores (such as re-painting the screen) that programmers are required to write for themselves in 3GL languages. In Clarion, a “Hello World” program is just 9 lines of source code, while in C/C++ you must write over 80!

### The Secret of Productivity

The Secret of Productivity (you’ve all heard this before) is *working smarter, not harder*. That is exactly what Clarion is designed to allow you to do with its Template-driven programming paradigm—Clarion writes the standard code for you so you don’t have to write it yourself, and then it is still flexible enough to let you add the “bells and whistles” (the fun part).

The secret to real programmer productivity without loss of flexibility is this: **always operate at the highest level of abstraction** that can get the job done the way it needs to be done. The “problem” with operating at higher abstraction levels is the loss of flexibility to always do things exactly the way you want. With Clarion, this problem is solved.

The Clarion toolset contains multiple abstraction levels—allowing you to always operate at the best abstraction level to do exactly what you need to do. You’re never “stuck” at a high level or forced to always “muck around” at a low level. The toolset generates object-oriented Clarion source code for you using Clarion’s *Application Builder Class (ABC) Library* (the high level), and when you need to, you can also add your own code (lower level) to augment the standard template-generated functionality.

Generally speaking, the higher the level of abstraction a programmer works at, the more productive they can be. At each higher abstraction level, the programmer becomes more productive by letting the computer do more of the work for them—generating standard code for standard tasks. This is the fundamental principle behind Clarion—and we implement this principle through Template-driven programming.

## Template Driven Programming

---

Clarion's Application Generator is "template driven." This means that it is a tool that changes based on the requirements of the template you are using to generate your code. Clarion's Procedure, Control, Extension, and Code Templates all write Clarion language source code for you, giving you a tremendous productivity boost. Clarion's templates provide many of the benefits of object-oriented programming, especially reusability, and the default template set generates truly object-oriented Clarion code for you using the *Application Builder Class (ABC) Library*. This makes Templates the real key in Clarion to **Rapid Application Development**.

### What is a Template?

In Clarion, a template is not just a "one-time" tool generating code that you must then maintain yourself (like the things that some other languages call "templates"), but is a **continually interactive tool** that asks specifically for the information it needs to generate your source code. Changing the information you provide to the template causes different source code to generate the next time you make your application. This interactivity makes Clarion's Templates the key to **Rapid Application Maintenance**.

Clarion templates allow you to insert your own Clarion source code into their generated source at any of the many Embed Points available to you within each template. This makes it unnecessary for you to maintain all the generated code in order to customize the procedure's behavior. It also guarantees that your special customizations aren't overwritten the next time you generate source—the template simply generates your code inside its standard code. This easy customization makes Clarion's Templates the key to **Rapid Application Enhancement**.

All the templates are stored in the registry file (REGISTRY.TRF). This file contains the pre-written executable code and data structures which you customize and reuse. You can use the Template Registry to modify the design of each template's default window or report to the way you want them to appear when you first create a procedure.

You can modify the Clarion templates. You may also add third party templates, or write your own, and use them *in addition to, and along with*, the Clarion templates. This makes the Application Generator an infinitely extensible tool.

### How do you use Templates?

When you create a new procedure, you identify the Procedure template that generates code that does the task you need to perform, then you customize it with the development tools. These Procedure templates include elements such as "browse windows" for viewing groups of records, and "form windows" for editing one record at a time. If the procedure is for a window

with a menu, the menu actions are automatically added to the application's procedure tree, and marked "ToDo," as any other procedure call would be.

The usual way to customize a procedure is to call one of the formatters. The Window and Report Formatters are visual design tools that allow you to "point and click" to design windows and reports. You just pick a control from a toolbox, CLICK to place the control, then RIGHT-CLICK to modify its properties.

Once you have created a procedure you can also use the Control, Extension, and Code templates to add even more functionality to the procedure. Control templates contain controls and generate all the standard executable code needed to use and maintain them. All the pre-populated controls that appear in the default window designs for the Procedure templates are actually Control templates that perform all the functions of the procedure.

Extension templates add executable code that increase the functionality of the procedure. Each typically provides you with on-screen instructions on what information to "fill in" to incorporate its functionality into the application.

Another way you may customize a procedure is to add your own embedded source code. The Application Generator displays a tree displaying all of the Embed Points that you have: before, during, and after the procedure's main logic, and for every event the window or controls in the procedure can generate. You can also directly edit your embedded source within the context of all the generated source code. This allows you to pick the precise logical point at which to execute the code, then "hand code" it, or tell a Code template to write code for you. The Application Generator generates all your application's source code from the templates and all your customizations (including embedded source code).

Clarion provides a rich assortment of standard templates with which you can rapidly develop applications. Just as the *Quick Start Tutorial* chapter of this book introduced you to the Wizards, the Application Generator tutorial in the *Learning Clarion* manual introduces you to using Clarion's Templates to produce any Windows application you need.

## Clarion's Levels of Abstraction

---

At this point in your introduction to Clarion, you've already worked with Clarion's highest abstraction levels: the Wizards. You'll work with most of the following features later in the tutorial in *Learning Clarion*.

### **Dictionary Editor and Data Modeller**

Your Data Dictionary is a single repository for all the data file definitions and their default data entry control settings. For existing data, defining your files can be as simple as importing the file's definition into the Data Dictionary directly from the data file itself!

**Application Wizard**

Generates a complete “standard” application for you in just minutes, based on the data file definitions and settings in your Data Dictionary.

**Procedure Wizards**

Generate “standard” procedures for you (like a Browse list, or data entry Form) based on the settings in your Data Dictionary.

**Procedure Templates**

Generate “standard” procedures for you (such as the application Frame menu, or a Report), based on the window or report design you create in Clarion’s Window Formatter and Report Formatter, and the options you select for the Template’s prompts for information about your procedure.

**Control Templates**

Add “standard” controls to your window or report design (such as a related files Tree control) and generate all the code needed to “drive” the behavior of the control.

**Extension Templates**

Generate additional code into a procedure to provide functionality unrelated to any specific window control (such as a Date/Time display in the application Frame’s status bar).

**Code Templates**

Generate code into a procedure that usually performs a single task related to a specific window control (such as Data Entry validation).

**Embedded Source Code**

You can write Clarion language source code of your own in any of the very numerous Embed Points in each procedure to customize and/or modify the behavior of the Template-generated code.

**Source Template**

A Template which allows you to completely write your own Clarion language procedures while still maintaining the application in the Application Generator.

**Windows API and C Standard Library**

In any Clarion language code that you write (whole procedures or into Embed Points), you can directly call Windows API functions or C standard library functions, if you really need to. A utility program will generate the Clarion language Windows API function prototypes, and C Standard Library prototypes are documented in the *Programmer’s Guide*.

**C/C++ or Modula-2 Code**

Write any TopSpeed C/C++ or Modula-2 functions that you need to and link them into your Clarion application. These compilers are included and are invoked by simply adding a .C, .CPP, or .MOD source code module to your Clarion project (also sold separately).

As you can see, the levels of abstraction available to you in Clarion span every level from the Application Wizard right down to writing C code (or even Assembly, if you want to work that hard).

## The Clarion Key to Maximum Productivity

---

The key to making Clarion work for you is to **always work at the highest abstraction level possible**:

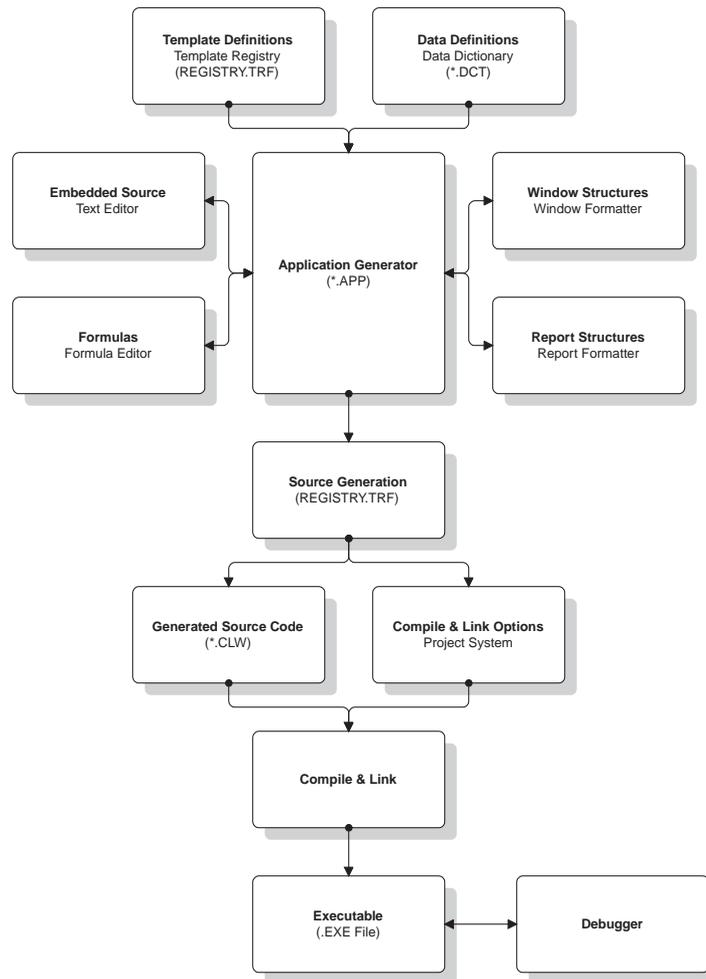
- Let the **Application Wizard** write a “starting point” application for you.
- Use the **Procedure Wizards** to add new procedures, as needed.
- Modify what you need to in each **Procedure Template** using Clarion’s two-way interactive tools (like the Window and Report formatters).
- Add **Control, Extension, and Code Templates** as you find you need their functionality.
- Customize it all with some **Embedded Source Code**.
- When you find you need to do something that no template does for you, use the **Source Template** and write it in the Clarion language (or get a third-party Template that does what you need).
- Dive all the way down to the **Windows API and C/C++** levels only when you absolutely must.

This is the Clarion philosophy of *working smarter, not harder*, by letting the toolset do the “drudge” work. Do this, and you’ll find that your productivity soars compared to any other tool you’ve ever used.

## Clarion's Development Environment

The development environment contains several main tools, all of which are accessible from the others. When using the Application Generator, buttons and menus in the various dialogs lead to the other tools.

The following Application Development Flow chart shows how all the tools common to both Professional and Enterprise Editions of Clarion interact with each other and the template registry, with the Application Generator at the center of the whole process:



This section provides a description of each tool, in the order that a typical programmer using the Application Generator might encounter them. Each tool contains dialog boxes which the programmer fills out to describe the Application's functionality to the Application Generator. On your command, the Application Generator generates the specified source code, and the Project System compiles and links it to make an executable program.

## Common Tools

---

Programming in Clarion is, in many ways, a personal journey through a series of dialog boxes. There is no mandatory sequence to follow through the dialogs, though some are prerequisites for others. The following are brief descriptions of the tools common to both Professional and Enterprise Editions of Clarion.

### **The Dictionary Editor**

The Data Dictionary (a .DCT file), maintained by the Dictionary Editor, holds a description of the database, including its files, keys, indexes, database drivers, file relations, fields, field validation rules, referential integrity constraints, and more. *This is the first file you create when you design your application.*

You can create the file definitions from scratch, or you can import definitions from existing data files. The other tools of the development environment use the information in the Data Dictionary to let you, for example, easily place data fields in a dialog box you design for the end user. The Application Generator creates code for all the statements that access the data files based on how you construct the Data Dictionary. In fact, the Application Wizard can generate a fully functional application based solely on your Data Dictionary!

### **The Application Generator**

The Application Generator generates your application's source code, one procedure at a time, based on the templates you pick from the template registry. It lets you add global and local memory variables, and customize the procedures with visual design tools and embedded source code.

The Application Generator provides access to the other tools of the development environment so you can customize the look and functionality of the windows, menus, reports and other user interface elements of your application. The templates provide their own controls (which appear within the Application Generator) allowing you to provide the information with which the template customizes the requested functionality.

### **The Window Formatter**

Visually design your application's windows and controls—everything the end user sees—in the Window Formatter. It automatically generates the source code for the elements you visually design on screen.

### **The Report Formatter**

The Report Formatter works with the Application Generator in much the same way as the Window Formatter. You place controls in a sample report

page. At run time, the print engine processes the records, handling page breaks, group breaks, headers, and footers as specified.

### **The Text Editor**

The Text Editor is a fully functional programmer's editor which you can use to write any source code your application needs. Most likely, when using the Application Generator, you'll use the Text Editor to create embedded source code to customize the way a procedure operates. Or, you can write entire applications from scratch (if you really want to work that hard).

The Text Editor features color coded syntax highlighting, making it easier to identify the different parts of the Clarion language statements for editing purposes. It also has full text search and replace capabilities, along with all the standard editing tools.

### **The Formula Editor**

The Formula Editor helps you quickly generate and manage simple or complex assignment statements based on any kind of mathematical or string expression. The Formula Editor provides syntax checking, plus instant access to all the variables, functions, and operators, so that the formulas generated are always syntactically correct.

### **The Project System**

The Application Generator automatically creates the project file (.PRJ) for the application for you. The project file contains compile and link options, such as whether to include debug code, optimization choices, external files, the source code files, libraries and other external files included in the compile and link process.

### **The Debuggers**

Debugging a program usually requires running the program and repeatedly stopping it at various points during execution to examine the value of different variables to determine the cause of logic errors in the program. The Clarion Debuggers (both 16-bit and 32-bit) have a number of windows which display source code, variable contents, active procedures, and much more.

There are three tutorials on using the Debuggers contained in the *User's Guide*. We recommend that you go through them after finishing all the tutorials in *Getting Started* and *Learning Clarion*.

### **The On-Line Help**

Clarion provides extensive on-line context sensitive help from almost every screen. Click on the **Help** button, or press F1.

The Common Questions section of the on-line help provides instructions and *examples* for performing common application tasks. You can cut and paste example code directly from the help system into your program. Choose **Help ► Contents**, then press the **How do I ... ?** button.

The Clarion on-line help system includes a main help file, plus auxiliary files. The on-line help files are searchable by keyword and by index. To perform keyword or index searches on an auxiliary file, open the file directly with the Windows Explorer or Program Manager.

## Enterprise Tools

---

The following are brief descriptions of the tools which are in the Enterprise Edition of Clarion and are not available in Professional Edition, except as add-ons (in certain cases). These tools are all fully documented in the *Enterprise Tools* manual—including tutorials for each.

### **Dictionary Synchronizer**

The Dictionary Synchronizer allows you to synchronize one data dictionary to another (available only in Enterprise Edition). You can synchronize two Clarion data dictionaries (.DCT files), or synchronize a Clarion data dictionary with an SQL backend's data dictionary (either direction). This allows you to get all the file definitions and relationships at once from the SQL backend database and eliminate a lot of front-end work and keep the Clarion data dictionary in synch with the SQL RDBMS at all times, no matter what changes the SQL Database Administrator makes.

### **Data Modeller**

The Data Dictionary (a .DCT file), can also be maintained by the Data Modeller (also available as an add-on to Professional Edition). This is a visual data modelling tool, allowing you to view your database design from an entity-relationship diagram perspective.

### **Version Control/Team Developer**

The Version Control/Team Developer tool allows you to checkpoint your applications and revert to previous versions, as needed (also available as an add-on to Professional Edition). In a team development environment, individual team members may check out just the portions of an .APP file that they need to update, allowing multiple developers to work on the same .APP at the same time without conflict. This uses PVCS on the backend to provide industrial-strength version control capability.

## Business Math Library

The Business Math Library is a set of standard business and statistical functions which you may use (also available as an add-on to Professional Edition). There are also a full set of templates which make implementing these functions in your programs very easy.

## Wise for Clarion

Wise for Clarion is a complete installation set creation tool from Wise Solutions, Inc. This is a special, Clarion-specific edition of the much-acclaimed Wise installation product line.

## Where to next?

---

So where should you go from here to learn more about programming with Clarion?

- The best place to go next is the *Learning Clarion* book. The two tutorials it contains are designed to introduce you to all the common development tools Clarion offers in both Professional and Enterprise Editions.
- TopSpeed offers educational seminars at various locations. Call Customer Service at (800) 354-5444 or (954) 785-4555 to enroll.
- Join (or form) a local Clarion User's Group and participate in joint study projects with other Clarion developers.
- Participate in TopSpeed's online forums. You'll find these on CompuServe (GO TOPSPEED) or on Internet newsgroups.

TopSpeed's internal news server offers newsgroups for all TopSpeed products. On this news server you can exchange ideas with other Clarion programmers as well as receive help from Team TopSpeed members and TopSpeed employees. Log into *tsnews.clarion.com* and subscribe to the groups for the products you want to discuss (**Strongly recommended!**).

In addition, there is a Usenet newsgroup—*comp.lang.clarion*— where you can network with other Clarion programmers from all around the world.

Good luck and keep going—the programming power that Clarion puts at your fingertips just keeps on growing as you learn more!

