
Chapter 2

Using the Runtime Executable

Introduction

This chapter explains how to use the Runtime executable (RRWRUN.EXE) to run R&R reports from the Windows Program Manager or from within Windows application programs. The explanation of the Runtime executable is presented in the following sections:

- ❑ Executing the Runtime
- ❑ Using Control Tables and Files
- ❑ Understanding the Runtime Status File
- ❑ Application Calls to the Runtime Executable
- ❑ Parameter Passing

As noted in Chapter 1, using the Runtime executable is one of three available methods for accessing the R&R Runtime. The other methods are explained in Chapter 3, “Accessing the Runtime DLL,” and Chapter 4, “Using the R&R Custom Control.”

Executing the Runtime

To use the Runtime executable to run an R&R report, follow these general steps:

1. In interactive R&R, create and save each report you want to run.
2. Using your database software or a text editor, create a Runtime control table or file that identifies the report and report library, as well as any parameters you want to change at runtime. The structure and contents of control tables and files are described in the **Using Control Tables and Files** section of this chapter.
3. Execute the Runtime in one of the following ways:
 - ◆ From the Windows Program Manager, select File ⇒ Run and enter the Runtime command line.
 - ◆ From the Windows Program Manager, select File ⇒ New and create a Program Item using the Runtime command line.

- ◆ Use the Quick Runtime Icons utility to create Windows Program Items for your reports so that you can access them simply by double-clicking an icon in the Program Manager.
- ◆ Include a call to the Runtime in your Windows application.

Providing Runtime Input

When you run an R&R report with the Runtime executable, you use either a database table (referred to as a *control table*) or a text file (referred to as a *text control file*) to specify each report you want to run, the library that contains each report, and any parameters you want to modify at runtime.

You create a control table using your database software. You can create a text control file using any database language, text editor, or word processor that produces unformatted text files. For details on control tables and files, see the **Using Control Tables and Files** section of this chapter.

The Runtime Command Line

After creating a report and a control table or file, you can use the Runtime to generate the report. When you call the Runtime, you must include the name of your control table or file and, optionally, one or more control table record numbers that identify the reports to be run.

Command Line Using a Control Table

If you use a database table to provide Runtime control parameters, the command syntax is:

```
RRWRUN <table name> [<record #> ...][switches]
```

Substitute the name (and optionally the path) of the control table for <table name>. For <record #>, substitute the record number(s) of the report(s) you want to run; if you do not include one or more record numbers, Runtime will execute all reports specified in the control table. In addition, you can optionally include one or more of the command switches listed in Figure 2.1.

For example, to generate the reports specified by records 1 and 2 of the RRWRUNIN.DBF control table, use a command like this:

```
RRWRUN RRWRUNIN 1 2
```

Command Line Using a Text Control File

If you use a text file to supply control parameters, the command syntax for running a single report is:

```
RRWRUN /T<file name> [switches]
```

Substitute the name (and optionally the path) of the text control file for <file name>. In addition, you can optionally include one or more of the command switches listed in Figure 2.1.

To run multiple reports using a text file, you must create a *command file*, an unformatted text file that lists the relevant text control files. First create a separate control file for each report you want to run; then create a command file listing the control files. To execute the Runtime with a command file, use the following syntax:

```
RRWRUN @<command file name> [switches]
```

For example, if you created three control files, you could then create a command file named REPORTS.CMD that lists these three control files (each on a separate line) and execute Runtime using the following command:

```
RRWRUN @REPORTS.CMD
```

You can optionally include one or more of the command switches listed in Figure 2.1.

Command Switches

The Runtime command can include one or more of the optional command switches listed in Figure 2.1. If you are using a text control file, the /T argument must appear first; the other switches can be in any order on the command line and can be either upper or lower case.

Switch	Specifies...
/T	Path and name of text control file
/R	Default report library directory
/D	Default data directory
/I	Default image file directory
/Xnnn	Default index file name extension
/W[Y N]	Whether database users can (Y) or cannot (N) modify database files while in use by the Runtime
/O	Status file name and/or directory
/E[X O]	Memo field type: (X) for an Xbase editor or (O) for another memo editor
/H	Suppresses printing of Title and Summary lines when no records are found.

Figure 2.1 Command Switches

These switches are explained in the following sections.

Text Control File Name (/T)

To execute Runtime using a text control file, include the name (and optionally the path) of the control file preceded by /T as the first argument to the Runtime command.

For example, the following command will run the report specified in the SALES.TXT control file:

```
RRWRUN /TSALES.TXT
```

Default Library Directory (/R)

To specify a default report library directory where the Runtime will look for the report library specified in the RI_LIBRARY control table parameter, use the /R switch in the Runtime command. The directory you specify with this switch will override any default library directory specified in the RRW.INI file.

For example, the following command specifies C:\LIB as the default library directory:

```
RRWRUN RRWRUNIN 1 2 /RC:\LIB
```

This command will run the reports specified in the first two records of the RRWRUNIN table. The Runtime will look for the R&R library file in C:\LIB.

Default Data Directory (/D)

The Runtime looks for the data files (tables, indexes, and text memo files) used by a report in the directory saved with that report. To specify a default data directory where the Runtime will look for data files when they are not in the saved directory, use the /D switch in the Runtime command. The directory you specify with this switch will override any default data directory specified in the RRW.INI file.

For example, the following command specifies C:\DATA as the default data directory:

```
RRWRUN  RRWRUNIN  1  2  /DC:\DATA
```

This command will run the reports specified in the first two records of the RRWRUNIN table. If the Runtime cannot find the data files in the location saved with the report, it looks in C:\DATA.

Default Image File Directory (/I)

To specify a default directory where the Runtime may look for image files used in the report, use the /I switch with the Runtime command. The directory you specify with this switch will override any default image directory specified in the RRW.INI file.

For example, the following command specifies C:\IMAGES as the default image directory:

```
RRWRUN  RRWRUNIN  1  2  /IC:\IMAGES
```

Default Index File Extension (/X)

If the index files used with the Runtime have a different file extension than the index files saved with a report, you can use the /X switch to specify a default index file extension. The Runtime will look for index files with the default extension if it cannot locate the index files saved with the report.

Syntax for this switch is **/Xnnn**, where **nnn** is the 1- to 3-character index file extension. The extension you specify with this switch will override any index file extension specified in RRW.INI.

For example, the following command specifies *ntx* as the file extension the Runtime should use when it cannot locate index files using their saved extensions:

```
RRWRUN  RRWRUNIN  1  2  /Xntx
```

File Write Access (/W)

When executing the Runtime in a multi-user or multi-tasking environment, you can use the **/W** switch to control whether database users can modify the tables and indexes in use by reports. Use the Y or N argument to the **/W** switch to control access to these files. The write access you specify with this switch will override any write access setting in RRW.INI.

To allow database users to modify files in use by a Runtime report, include the switch **/WY** in the Runtime command. This setting might not be appropriate for all reports, but it is useful when you want to allow database users to update files while you are reporting on them.

If you do *not* want users to modify files in use by a report, include the switch **/WN** in your Runtime command to prevent users from modifying tables and indexes while the report is running. Other R&R users will still have access to those files. (For more information about file write access settings, see the explanation of the **Allow Other Users to Update Database Tables** setting in Chapter 6, “Setting Defaults,” in *Using R&R*.) Note that this switch controls R&R’s behavior only when accessing shared data; your database or network software may impose other file access restrictions.

Status File Name (/O)

You can distinguish Runtime status files by using the **/O** switch to specify the directory in which the file will be created and/or to specify the complete status file name.

For example, the following command generates a status file named MYSTATUS. Because no path is specified, the status file will be created in the current directory.

```
RRWRUN  RRWRUNIN  5  /OMYSTATUS
```

To specify the directory in which a status file should be created, enter a full path and name. If you enter a path without a file name, the Runtime will create a table named RRUNOUT.DBF (if you are using a control table) or a text file named RRUNOUT.OUT (if you are using a text control file) in the specified directory.

Memo Editor (/E)

To identify the type of memo editor that was used to create database memos, use the **/E** switch. Use **/EX** to indicate that memos were created with an Xbase memo editor or **/EO** to indicate that memos were created with any other memo editor.

Title/Summary Lines for No Records Found (/H)

By default, the Runtime will print Title and Summary lines even when no records are found. To cause the Runtime to print nothing (no Title or Summary lines) when no records are found, execute the Runtime with the **/H** switch.

Using the Runtime on a Network

Since the Runtime is available for unlimited use, you need not purchase more than one copy of R&R to enable multiple users to run Runtime reports. More than one user can run Runtime reports on a network with only a single copy of the Runtime programs stored on the network server. Use the Runtime commands in the previous section to run reports from within an application or from Windows.

Note that the Runtime control table is open for shared access when the Runtime program is active on a network. In some applications, it may be possible for two users to access the same record in the control table at the same time. For example, two users might specify query expressions that affect the value of the RI_FILTER parameter in the control table. In this case, the Runtime would generate two reports using the second query expression. To better manage multiple-user access to the control table, create a separate record in the table for each user. You can also use the **/O** switch to create a unique output status table for each user.

Using RRW.INI for Default Information

If RRW.INI, the R&R configuration file, is in the Windows directory, the Runtime will use the default settings specified in that file. However, command-line switches take precedence; any setting you specify using a command-line switch will always override the corresponding RRW.INI setting.

Figure 2.2 lists the RRW.INI settings that the Runtime will use (unless a command-line switch is used instead).

RRW.INI Setting	Specifies
DataDir	Default data directory
ImgDir	Default image file directory
ImgExt	Default image file extension
LibDir	Default library file directory
MemExt	Default memo file extension
IndExt	Default index file extension
AllowW	Database file write access
MemoEd	Default memo editor
PrevWinClr	Preview window color

Figure 2.2 RRW.INI Settings Used by Runtime

Using Control Tables and Files

The following sections describe the structure and contents of database control tables and text control files.

- ❑ Creating a Control Table
- ❑ Creating a Text Control File
- ❑ Specifying Control Parameters

Creating a Control Table

Use your database software to create a table that contains columns for the control parameters; then add a row for each report you want to run and enter values in the appropriate columns. In the simplest case, the table can include just the name of the report library file (as the RI_LIBRARY value) and the report name (as the RI_REPORT value). When these values are specified, the Runtime outputs the report to the destination saved with the report.

Follow these guidelines in creating a control table:

- ❑ Specify each parameter in a separate column; the column name must be the same as the parameter name.
- ❑ Predefined parameters must use the column names and data types specified in Figures 2.3 and 2.11. For character columns, use any supported character data type. For numeric parameters, use any

numeric data type, but note that the numeric parameters accommodate only integer values.

- User-defined parameters can use any column name and supported character data type.
- Parameters can be in any order.
- The only required parameters are RI_REPORT (or, alternatively, RI_REPPICK) and RI_LIBRARY.
- You can omit unused (blank) parameters.
- Specify parameters for each report in a separate row.

Creating a Text Control File

You can also create a control file using any text editor or word processing program that saves unformatted text files. In the file, you specify the report name, library, and any optional parameter values. When you use a text file to control the Runtime, you must create a separate file for each report you want to run. However, you can run multiple reports with a single Runtime command by creating a command file that lists each control file.

The format for each parameter name and value in a control file is:

```
<parameter name>=<value>
```

Follow these guidelines in creating a text control file:

- Specify each parameter and its value on a separate line.
- Each parameter and its value must fit on a single line.
- The maximum length of a line is 1000 characters.
- You can list parameters in any order.
- Parameter names are case insensitive (that is, you can enter them in upper, lower, or mixed case).
- Predefined parameters must use the names listed in Figures 2.3 or 2.11; user-defined parameters can have any name. See the section in this chapter entitled **Parameter Passing** for information about user-defined parameters.
- RI_REPORT (or, alternatively, RI_REPPICK) and RI_LIBRARY are the only required parameters.
- Leading and trailing white space in both the parameter name and the value is ignored.
- Lines beginning with a left square bracket ([) are ignored.
- Lines beginning with a semicolon are ignored.

Specifying Control Parameters

The control table or file can contain two kinds of parameters:

- Predefined parameters specifying values that control frequently changed report features such as queries. The control table *must* contain columns for two of these parameters, RI_REPORT (or RI_REPPICK) and RI_LIBRARY. It can contain values for some or all of the other predefined parameters. These parameters are listed in Figures 2.3 and 2.11.
- User-defined parameters specifying values that control other report features such as user-supplied text strings, for use by the RIPARAM() function in the report. The control file need not include any user-defined parameters. If any of these parameters are present, the control file can contain values for some or all of them. For information about adding columns to the control file for user-defined parameters, see the **Parameter Passing** section of this chapter.

If the control file does not contain values for the two required parameters that indicate the report library and report name, the Runtime cannot run the report. If the control table does not contain values for any predefined or user-defined parameters designed to change characteristics of the saved report, the Runtime will run the report as saved.

Width of Predefined Parameters

Predefined parameters have maximum widths specified in Figure 2.3 and 2.11. While you should not exceed these widths, you can decrease the widths of these parameters to correspond to the actual width of your data. For example, if you plan to specify a scope using strings that will be no longer than 20 characters, you can define the RI_LOSCOPE and RI_HISCOPE parameters as 20 characters wide. User-defined parameters can be up to 512 characters wide.

Parameter Values

Parameters that require character values can contain upper, lower, or mixed case letters, unless the parameters contain values used in a query. By default R&R is case insensitive, but if you edited RRW.SRT to make R&R case sensitive, you should enter query values in the case used in the database.

Some predefined parameters can have a question mark (?) value in the control table or file. Use the question mark to specify that the Runtime

should display a dialog box prompting the user to enter or select a value. For example, when RI_PRINTER contains a question mark, the Runtime will display a dialog box prompting the user to choose screen, printer, or export as the report's output destination.

The question mark parameter value is explained in more detail in the descriptions of the parameters for which it is valid: RI_PRINTER, RI_REPPICK, RI_WPTR, RI_WPORT, RI_SCOPE, RI_QUERY, and user-defined parameters.

Parameters for Modifying Report Characteristics

Figure 2.3 lists the predefined Runtime control parameters that can be used to control report characteristics. The next section of this chapter lists and explains the parameters that apply specifically to the size and appearance of the preview window at runtime.

Each parameter name has the prefix RI_, indicating that it provides **R**untime **I**nput. In the Data Type column, the letter **A** represents the datetime data type, **C** represents the character data type, **D** represents date, **N** represents numeric, and **L** represents logical.

Parameter Name	Contents	Data Type	Max. Width
RL_ALIAS1 – RI_ALIAS99	In each, a related table, index, or index tag name	C	300
RL_BEGPAGE	Beginning page number	N	9
RI_CHKTIME	Checkpoint frequency flag	C	1
RL_COPIES	Number of copies	N	9
RL_DISPERR	Display-error flag	C or L	1
RL_ENDPAGE	Ending page number	N	9
RL_EXPDST	Export destination	C	8
RL_FILTER	Query expression	C	1024
RI_GROUP1 – RI_GROUP8	Group field override	C	20
RL_HISCOPE	High scope value	C	250
RL_LIBRARY	Report library name	C	128
RL_LOSCOPE	Low scope value	C	250
RL_MASTER	Master table name	C	128
RI_MEMO	Text memo file name	C	128
RI_MINDEX	Master index information	A,C,D, or N	150
RL_NOESC	User escape flag	C or L	1
RL_OUTFILE	Output file name	C	128
RI_PRINTER	Destination	C	32
RL_QUERY	Query flag	C	1
RL_REPORT	Report name	C	30
RI_REPPICK	Report select/pick flag	C	1
RL_SCOPE	Scope flag	C	1
RI_SORT1 – RI_SORT8	Sort field override	C	21
RL_STATUS	Display-status flag	C or L	1
RL_TEST	Test-pattern flag	C or L	1
RL_WPORT	Printer port	C	40
RL_WPTR	Printer name	C	40
RL_WTITLE	Window title	C	200

Figure 2.3 Predefined Runtime Control Parameters

The following sections explain the values required by each parameter.

RI_ALIAS1 – RI_ALIAS99

These parameters are optional. You can use each of the parameters to specify a related table, index, and/or index tag name to override those saved with the report. The syntax for a file override specification is:

```
<alias> = <table>,<index>,<tag>
```

In this specification, **<alias>** represents the R&R alias assigned to the table in the saved report; **<table>** represents the replacement table, which can be a complete path and table name, a directory, or a table name; **<index>** represents the name and optional path of the new index file; and **<tag>** represents the name of an index tag if the specified index file is a multiple-field index file. You must specify at least one of the three replacement names in any override specification; you can specify all three.

For example, the following specification replaces the related table assigned the CUST94 alias in the saved report with a table named CUST95.DBF in the DATA directory on drive C. It also replaces the saved index file with the CUST95.MDX index file, and the saved index tag with the CUSTID index tag:

```
CUST94 = C:\DATA\CUST95,C:\DATA\CUST95.MDX,CUSTID
```

You can also use a file override specification to replace just the index file, just the index tag, or both. If you omit the **<table>** or **<index>** part of the specification, you must use a comma as a place holder.

For example, the following specification replaces the index file used to read the CUST95 table with the LASTNAME.NDX file in the DATA directory on drive C. The comma serves as a place holder for the **<table>** specifier:

```
CUST95 = ,C:\DATA\LASTNAME.NDX
```

For a multiple-field index, the following override specification replaces the saved tag used with the CUST94 table and index with the LASTNAME tag:

```
CUST95 = ,,LASTNAME
```

If you do not include file override specifications in any of these fields, the Runtime uses the database and index files saved with the report. It searches for these files using the search rules explained in Chapter 7, “Distributing Reports.”

RI_BEGPAGE, RI_ENDPAGE

These parameters are optional. The beginning and ending page number parameters allow you to override the starting and ending page numbers saved with the report. The default value for these parameters is blank.

To specify page numbers, include an RI_BEGPAGE value, an RI_ENDPAGE value, or both. If you specify both, RI_ENDPAGE must be equal to or greater than RI_BEGPAGE. For example, users can restart a canceled report where it was interrupted by specifying the starting page number as the RI_BEGPAGE value. (See the description of the RO_PAGES field in the **Understanding the Runtime Status File** section.) To reprint one or more consecutive pages of a report, specify the page numbers in the RI_BEGPAGE and RI_ENDPAGE parameters. To print just one page, specify the same page number for both parameters.

RI_CHKTIME

This parameter is optional. The checkpoint frequency flag controls how frequently the Runtime status table, by default RRUNOUT.DBF, is updated. The checkpoint flag can contain the letter **R** or **P**. **R** tells the Runtime to update the RRUNOUT table after completing each report; **P** tells the Runtime to update the table after completing each page. The default value is **R**.

Specify **P** as the checkpoint value if you want Runtime users to be able to determine how much of a report was printed before an abnormal termination (for example, a system failure). When this value is **P**, the Runtime will update the RO_PAGES page number value in the status table after each page of the report is processed. (See the section entitled **Understanding the Runtime Status File**.) In case of a report termination, the report can be restarted where it left off.

If your application doesn't require the ability to restart terminated reports, specify **R** and the report will print a bit faster. Users can always reprint a report starting at the beginning.

RI_COPIES

This parameter is optional. It contains the number of copies of the report you want to print. The number must be between 0 and 999, inclusive. If you leave this parameter blank or enter 0, the Runtime prints the number of copies saved with the report.

RI_DISPERR

This parameter is optional. It controls whether errors encountered by the Runtime are displayed on the screen. If the parameter contains a true logical value (**T**), any Runtime error messages are displayed in addition to being written to the Runtime status table, by default RRUNOUT.DBF. If the parameter contains a false logical value (**F**) or is blank, the Runtime error messages are not displayed, but are written to the Runtime status table. If the Runtime cannot open the status table, an error message is displayed regardless of the RI_DISPERR value.

RI_EXPDST

Use this parameter to specify the destination (display, file, or printer) for a report saved with an Export Type setting of Excel Pivot Table or Excel Chart (you must also specify the appropriate value in RI_PRINTER). A value of **D** will cause Excel to display the pivot table or chart; **F** will cause Excel to send it to the file specified by RI_OUTFILE; and **P** will cause Excel to print it to its default printer.

RI_FILTER

The optional RI_FILTER parameter contains a logical expression that will override the query saved with a report, if any, when the value in RI_QUERY is **O** for Override.

RI_FILTER expressions use the same syntax as calculated field expressions that return logical values. (For details, see Chapter 8, “Working with Calculated Fields,” in *Using R&R*.) The RI_FILTER expression can be up to 1024 characters long. When an expression is specified and the value of RI_QUERY is **O**, the Runtime selects all records where the value of the RI_FILTER expression is true. The expression can refer to any data available in the report, as well as many calculated and total fields.

For example, if you enter the expression **CITY="Dallas"**, the Runtime will select all records where the value of this expression is true, in other words all records where the value in the CITY field is Dallas. If the city name is in a memo field named NOTE, the expression **NOTE="*Dallas*"** will select all records in which the NOTE field contains the word “Dallas.”

Entering the expression **PASTDUE=T** tells the Runtime to select all records where the value in the PASTDUE field is the logical true value. Entering **AMOUNT>=200** will select all records where the value in the AMOUNT field is equal to or greater than 200. Entering the following

expression will select all records where the date in the INVDATE field of the RRORDERS table is January 31, 1995:

```
RRORDERS->INVDATE={01/31/95}
```

You can enter compound expressions using parentheses. For example, the following expression selects all records where the value in the CITY field is either Dallas or Houston and where the value in the SALES field is greater than 50,000:

```
(CITY="Dallas" or CITY="Houston") and SALES>50000
```

Certain restrictions apply to the use of total fields in RI_FILTER expressions. The totals must be pre-processed. If the totals are group totals, you can use only the highest-level preprocessed total in the RI_FILTER expression. For example, if the report has pre-processed subtotals at group levels 1 and 2, the RI_FILTER expression can contain only the level-1 pre-processed subtotal.

RI_GROUP1 – RI_GROUP8

The optional RI_GROUP parameters (RI_GROUP1 through RI_GROUP8) enable you to specify different group fields from those saved with the report. Figure 2.4 explains the possible values for these parameters (in each case, substitute the table alias for “alias” and the field name for “fieldname”).

Value	Changes Group Selection to
alias->fieldname	Field <i>fieldname</i> in table <i>alias</i>
fieldname	Field <i>fieldname</i> (<i>fieldname</i> must be unique)

Figure 2.4 Group Field Override Values

You must specify group overrides beginning with the first level you want to change and proceeding to the depth desired (that is, you cannot skip group levels).

RI_LIBRARY

This parameter is required. It identifies the library that contains the report. The library name can include a path. The .RP5 extension is optional. For example, a value of C:\DBASE\DATA\CUSTOMER in this parameter identifies the report library as CUSTOMER.RP5 in the subdirectory \DBASE\DATA on drive C.

If you don't include a path, the Runtime searches for the file in the default library directory specified on the command line or in RRW.INI. If no default is specified on the command line or in RRW.INI, the Runtime searches for the library in the current directory.

If you leave this parameter blank or if the library you specify cannot be found or read, the Runtime writes an error in the status table and, optionally, displays an error message box (see RI_DISPERR).

RI_MASTER

This parameter is optional. It contains the name and/or directory location of a table that will override the master table saved with the report. The fields in the master table you specify with RI_MASTER must match in name, number, and data type the fields in the original master table.

- If you specify both a directory and a table name, this directory is the only directory searched and this table name is the only table the Runtime searches for.
- If you specify a directory without a table name, the Runtime searches the specified directory for the master table name saved with the report.
- If you specify a table name without a directory, the Runtime searches for a table with the specified name in the directory of the master table saved with the report, then in the default data directory specified on the command line or in RRW.INI. If no default data directory is specified, the Runtime searches for the table in the current directory.

If you omit this parameter or leave it blank, the Runtime uses the master table saved with the report.

RI_MEMO

This parameter is optional. It contains the name and/or directory location of the text memo file used in the report, which will override the text memo file saved with the report (if any).

- If you specify both a directory and a file name, this directory is the only directory searched and this file name is the only file the Runtime searches for.
- If you specify a directory without a file name, the Runtime searches the specified directory for the text memo file name saved with the report.

- ❑ If you specify a file name without a directory, the Runtime searches for a file with the specified name in the directory saved with the report, then in the default data directory specified on the command line or in RRW.INI.

If you leave this parameter blank, the Runtime uses the text memo file saved with the report, if any.

RI_MINDEX

The RI_MINDEX parameter enables you to specify a master index for a report that was saved without one, override the master index saved with a report, or remove a saved master index from a report. To specify a master index or override the saved one, the specification can consist of any or all of three values in the following order:

`<index>,<index type>,<tag>`

In this specification, <index> represents the index file specification, which can be a complete path and file name, a directory, or a file name.

- ❑ If you specify both a directory and a file name, this directory is the only directory searched and this file name is the only file the Runtime searches for.
- ❑ If you specify a directory without a file name, the Runtime searches the specified directory for the master index name saved with the report.
- ❑ If you specify a file name without a directory, the Runtime searches for a file with the specified name in the directory of the master index saved with the report, then in the current master database directory, then in the default data directory specified on the Runtime command line or in RRW.INI. If no default is specified, the Runtime searches for the file in the current directory.

In this specification, <index type> is the data type of the new index, represented as **A** for datetime, **C** for character, **D** for date, or **N** for numeric. You can omit this value if you are overriding a saved master index and the replacement index is the same type (although it is good practice to include the data type specifier). If the index named in the specification is a multiple-field index file (MDX, CDX, or WDX), the <tag> part of the specification represents an index tag. If you omit the <index> or <index type> part of the specification, use a comma as a place holder for each.

For example, the following specification replaces the master index saved with the report with an index named CUST95.MDX in the CUSTOMER directory on drive C. Since the replacement index is the same data type as

the original index, the specification includes a comma in place of the <index type>. It also replaces the saved index tag with the CUSTID index tag:

```
C:\CUSTOMER\CUST95.MDX,,CUSTID
```

The following specification changes only the index tag of the master index saved with the report:

```
,,CUSTID
```

To remove the master index saved with the report without specifying a new master index, use the RI_MINDEX value ,R.

If you omit this parameter or leave it blank, the Runtime uses the master index saved with the report, if any.

RI_NOESC

This parameter is optional. The user escape flag can contain either a true (T) or false (F) logical value. True means the Cancel button in the status window is not active while reports are being output. False means the user can select Cancel during report output to pause or end the job. The default value is false. Note that the status window appears only when RI_STATUS is set to true.

If the user cancels the report, the RO_ECODE field in the status table contains a C (see the section entitled **Understanding the Runtime Status File**).

RI_OUTFILE

This parameter is optional. It contains the name of an output file. Use it to send report output to a file, or use it in combination with RI_PRINTER and/or RI_EXPDST to export to any export type saved with the report. To send the report directly to the saved destination, omit this parameter or leave it blank.

- When RI_PRINTER is empty or contains the D or question mark (?) value, the Runtime outputs the report (including printer codes) to the file specified in RI_OUTFILE .
- When RI_PRINTER contains A, X, or W, the Runtime exports the report to the file specified in RI_OUTFILE (overriding the saved file name) as a text file (without printer codes), Xbase file, or worksheet file.
- When RI_PRINTER contains CSV, MSWORD, or RTF, the Runtime exports the report to the file specified in RI_OUTFILE

(overriding the saved file name) as a text data (comma-, tab-, or character-separated) file, Word Merge file, or Rich Text Format file.

- ❑ When RI_PRINTER contains **Excel Chart** or **Excel PivotTable** and RI_EXPDST is **F** (for file), the Runtime exports the report to the Excel file specified in RI_OUTFILE.

The name of the output file can include a path. For example, to send a report to a text file INVOICE.TXT in the C:\PROJECT\TEXT subdirectory, specify the following value for the RI_OUTFILE parameter:

C:\PROJECT\TEXT\INVOICE.TXT

If RI_OUTFILE does not include a path, the Runtime places the file in the current directory.

RI_PRINTER

This parameter is optional and can have one of the following values: **D**, **A**, **P**, **Excel Chart**, **Excel PivotTable**, **RTF**, **CSV**, **MSWORD**, **W**, **X**, or a question mark (?).

The **D** value specifies that the report be sent to the display, allowing the user to preview the report before printing it. After previewing the report, the user can select Print on the Preview screen to send the report to the printer saved with the report or specified as the RI_WPTR value. Note that if the value of RI_PRINTER is **D** and RI_OUTFILE is specified, the report will be output to the file specified in RI_OUTFILE when the user selects Print in the Preview screen.

The **A** value specifies that the report be sent to the text file named as the RI_OUTFILE value. The Runtime will export the report as a text file without printer codes.

The **P** value specifies that the report be sent to the printer saved with the report or specified as the RI_WPTR value, even if the report's saved destination is a file.

The **Excel Chart** and **Excel PivotTable** values specify that the report be exported to an Excel chart or Excel pivot table, respectively. If you specify one of these values, you can also include a value for RI_EXPDST to control the output destination (display, file, or printer).

Use the **CSV**, **MSWORD**, or **RTF** value to export to a text data file, Word Merge file, or Rich Text File, respectively. You can also specify an RI_OUTFILE value to override the output file name saved with the report.

To output a report to a worksheet or Xbase file, specify **W** or **X**, respectively, as the RI_PRINTER value and specify the file name as the RI_OUTFILE value. If you do not specify a file extension, R&R appends .WKS to worksheet files and .DBF to Xbase files.

The question mark (?) value allows the user to select the print destination (screen, printer, or export) at runtime. When the value of RI_PRINTER is a question mark, the user will see the dialog box shown in Figure 2.5. If RI_WTITLE is specified, the title bar will contain the RI_WTITLE value. If RI_WTITLE is empty, the title bar will contain the report name.

The user can select Screen to preview the report, Printer to print it, or Export to export it. If RI_OUTFILE contains a file name, the report will be output to the file specified by the RI_OUTFILE value if the user selects Export.

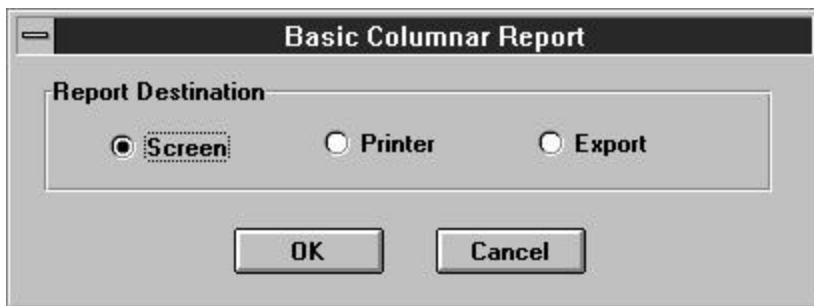


Figure 2.5 Print Destination Dialog Box

If you omit this parameter or leave it blank and RI_OUTFILE is empty or missing, the Runtime outputs the report to the printer saved with the report or specified as the RI_WPTR value. If you omit this parameter or leave it blank and RI_OUTFILE contains a file name, the Runtime outputs the report to a file with printer codes.

RI_QUERY

The optional RI_QUERY parameter allows you to control whether a query is applied to the report. RI_QUERY can have one of four values:

- ❑ **S** (Saved) means to run the report using the query saved with it, if any. The Runtime will ignore the expression in RI_FILTER and run the report exactly as it was saved.
- ❑ **E** (Entire) means to ignore any query saved in the report or contained in the RI_FILTER parameter.

- ❑ **O** (Override) means to override the saved query (if any) with the expression in the RI_FILTER parameter. The Runtime will generate the report with the records selected by the RI_FILTER expression.
- ❑ **?** (Question mark) means to allow the user to enter a query or edit the saved query at runtime. If no query was saved with the report, the Insert Selection Rule dialog displays, as shown in Figure 2.6.

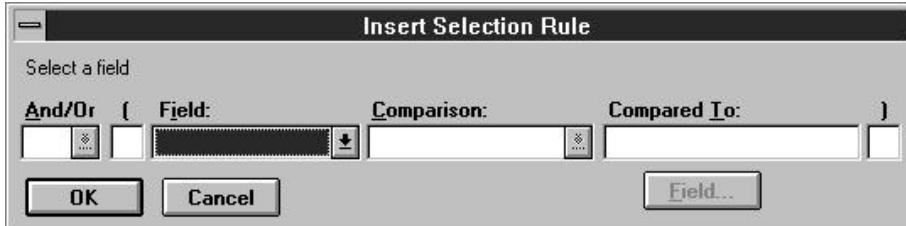


Figure 2.6 Insert Selection Rule Dialog Box

If a query was saved with the report, the Query dialog box displays, as shown in Figure 2.7.

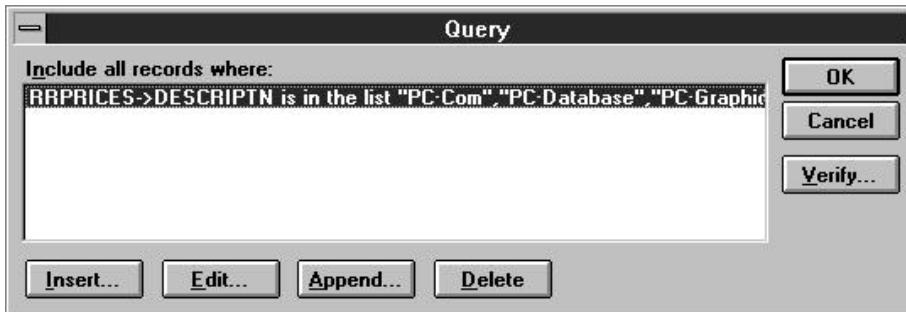


Figure 2.7 Query Dialog Box

When RI_QUERY contains a question mark (?), the value of RI_FILTER is always ignored.

RI_REPORT

This parameter is required (unless a value of **R** or **?** has been supplied for RI_REPPICK). It contains the name under which the report was saved. For example, to run a report named “Order Invoice,” enter **Order Invoice** in this parameter. The report you select must be in the library specified by RI_LIBRARY. Except for case, you must enter the name exactly as it was saved in your report library.

If you leave this parameter blank or if the report you select cannot be retrieved, the Runtime writes an error in the status table and, optionally, displays an error message box (see RI_DISPERR).

RI_REPPICK

This parameter is optional and can contain one of two values: **?** or **R**. If you include this parameter, you do not need to include the RI_REPORT parameter; if you include both RI_REPPICK and RI_REPORT values, Runtime ignores the RI_REPORT value.

Use the question mark (?) value in this parameter to have the Runtime prompt the user to select a succession of reports. When the value is a question mark (?), Runtime will prompt the user to select a report from the library specified in RI_LIBRARY. After Runtime executes the selected report, the user will then be prompted to select another report. This prompt for report selection will repeat after each report until the user selects Cancel.

Use the **R** value in this parameter to prompt the user to select just one report from the library specified in RI_LIBRARY. When the value is **R**, Runtime will prompt the user to select a report (as with the ? value), but will not prompt for an additional report selection after the report has been executed.

RI_SCOPE, RI_LOSCOPE, RI_HISCOPE

These parameters are optional. The scope flag, RI_SCOPE, allows you to control the range of master table records that should be included in the report. You can specify a range of record numbers or index key values, ignore the scope saved with a report, or prompt the user to enter a range at runtime. When you specify scope values, the Runtime reads only the records in the master table whose record number or index key is within the specified range. You can often speed up a report by using scope values with a master index. If you omit this parameter or leave it blank, the Runtime uses the saved scope values.

RI_SCOPE can contain one of four values:

- S** for “Saved,” which means to use the scope values saved with the report.
- E** for “Entire,” which means to ignore any scope values.
- O** for “Override,” which means to override the saved scope values with the values in the two parameters RI_LOSCOPE and

RI_HISCOPE. (Be careful to use the letter **O** and not the digit zero, 0.)

- Question mark (?), which allows the user to enter or change scope values at runtime. When RI_SCOPE contains a question mark, the dialog box shown in Figure 2.8 displays. If RI_WTITLE is specified, the title bar will contain the RI_WTITLE value. If RI_WTITLE is blank or missing, the title bar will contain the report name.

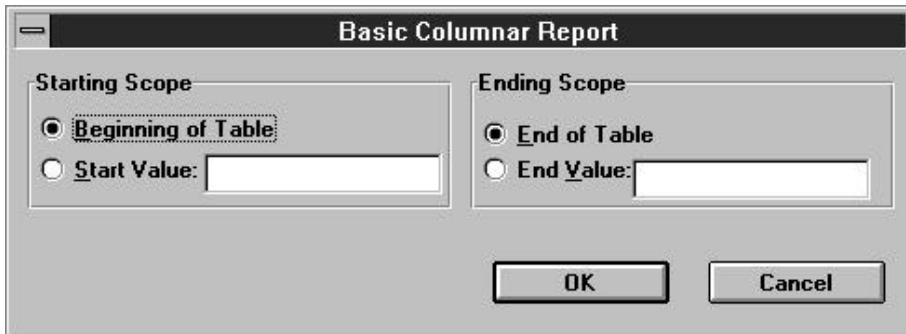


Figure 2.8 High and Low Scope Dialog Box

If RI_SCOPE contains **O** (Override), RI_LOSCOPE specifies the starting value of the scope and RI_HISCOPE specifies the ending value of the scope. If RI_SCOPE contains **S**, **E**, or **?**, the Runtime ignores RI_LOSCOPE and RI_HISCOPE.

Each RI_LOSCOPE and RI_HISCOPE value can contain either a record number or an index key value up to 250 characters wide. If no master index was saved with the report (or added using RI_MINDEX), the Runtime assumes the value is a record number. Otherwise, the Runtime assumes the value is a key value in the master index. In this case, the report begins reading the master table at the first record equal to or greater than the RI_LOSCOPE value and stops reading the master table after the last record found that is equal to or less than the RI_HISCOPE value.

The range fully includes the end points. In other words, if you enter A as the low value and M as the high value, the Runtime reads the first record in which the value begins with A through the last record in which the value begins with M. For example, if you have a customer table indexed on last name and you want to print invoices for all customers whose name begins with a letter between A and M, enter O in RI_SCOPE, the letter A in RI_LOSCOPE, and the letter M in RI_HISCOPE.

All scope values must be character strings. Note that a date scope value must be in the format mm/dd/yy or mm/dd/yyyy. Do not enclose scope values within quotes.

RI_SORT1 – RI_SORT8

The optional RI_SORT parameters (RI_SORT1 through RI_SORT8) enable you to specify different sort fields from those saved with the report. Figure 2.9 explains the possible values for these parameters (in each case, substitute the table alias for “alias” and the field name for “fieldname.”).

Value	Changes Sort to
+alias->fieldname	Field <i>fieldname</i> in table <i>alias</i> , ascending
-alias->fieldname	Field <i>fieldname</i> in table <i>alias</i> , descending
alias->fieldname	Field <i>fieldname</i> in table <i>alias</i> , ascending
+fieldname	Field <i>fieldname</i> , ascending (<i>fieldname</i> must be unique)
-fieldname	Field <i>fieldname</i> , descending (<i>fieldname</i> must be unique)

Figure 2.9 Values for RI_SORT Parameters

You must specify sort overrides beginning with the outermost sort field and proceeding to the last level you want to override (that is, you cannot skip sort levels).

RI_STATUS

The RI_STATUS parameter enables you to specify whether the Runtime should display a status window while it is generating a report. If the parameter contains a true logical value (T), the Runtime will display a status window. If RI_STATUS is set to true and RI_NOESC is set to false, the status window will contain a Cancel choice that allows the user to terminate a report in progress.

If the RI_STATUS parameter is missing, empty, or contains a false logical value (F), the Runtime will not display a status window; instead it will display as an icon while it is running.

RI_TEST

This parameter is optional. The test pattern flag can contain either a true (**T**) or false (**F**) logical value. True means to display a dialog box before printing the report to allow the user the option of printing a test pattern. False means don't offer a choice to print a test pattern. If the parameter is blank, the user is not offered the choice of printing a test pattern.

A test pattern is useful for aligning forms in the printer. The user can print the test pattern as many times as necessary and then print the report. If you enter **T**, the Runtime displays a box containing OK, Cancel, and Print buttons. The user can select OK and print as many test patterns as necessary to align the forms. Once the forms are aligned, the user can select Print to begin printing the actual report.

Note that a test pattern includes only page header, record, and page footer lines.

RI_WPORT

This parameter is optional. Enter a value such as "LPT1:" to override the printer port (and the printer associated with that port) saved with the report. Note that the colon is required. If both RI_WPTR and RI_WPORT values are supplied, they must match an installed Windows printer.

You can also use the question mark (?) value for this parameter. When RI_WPORT contains a question mark, the user will see the Printers dialog box shown in [Figure 2.10](#). See the description of the RI_WPTR parameter.

RI_WPTR

This parameter is optional. Enter a value to override the printer saved with the report. This parameter can have one of two values:

- The name of an available Windows printer (for example, "HP LaserJet Series III"). Available Windows printers are listed in the R&R Printers dialog (accessed from File ⇒ Printers in interactive R&R). The value is case insensitive. If you enter a value in this parameter and RI_WPORT is blank, R&R uses the port associated with the printer name in the list of available Windows printers.
- The question mark (?) value, to allow the user to select a printer at runtime. When RI_WPTR contains a question mark, the Printers dialog displays, as shown in [Figure 2.10](#).

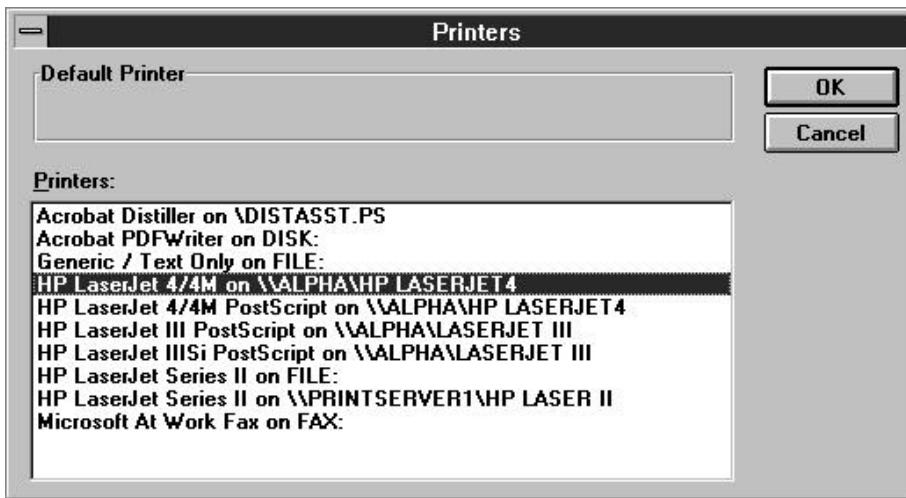


Figure 2.10 Printers Dialog Box

Printer Setup in the Windows Control Panel controls which printers are listed in the Printers dialog box. R&R initially highlights the printer saved with the report. The user can select another printer and port as necessary.

If this parameter is blank, the Runtime uses the printer saved with the report. If the report was saved with the “Print to File” option selected and the value of RI_PRINTER is blank, the RI_WPTR value controls which printer driver the Runtime will use. If the value of RI_PRINTER is A, CSV, MSWORD, RTF, W, or X for export to a file, the Runtime will ignore the value in RI_WPTR.

RI_WTITLE

This parameter is optional. Use this parameter to specify a report title (for example, “Quarterly Profits”) that will display in the following places:

- The Title Bar of the Preview window;
- The Print Status window (if RI_STATUS = T);
- Below the Runtime icon (if RI_STATUS = F);
- The title bar of the dialog box that displays when a question mark is specified as the value for RI_REPPICK, RI_PRINTER, RI_SCOPE, or any user-defined parameter.

If this parameter is blank, the Runtime will use the report name as the window title.

Parameters to Control Runtime Preview Display

The parameters listed in Figure 2.11 control the size and appearance of the preview window at runtime.

Parameter Name	Controls	Data Type	Max. Width
RI_WBORDER	Presence and type of border	N	1
RL_WCTRL	Presence of system control box in caption bar	L	1
RL_WHEIGHT	Height of preview window in pixels	N	4
RL_WLEFT	Left position of preview window in pixels	N	4
RL_WMAX	Presence of maximize button in caption bar	L	1
RL_WMIN	Presence of minimize button in caption bar	L	1
RL_WPARENT	Assignment of parent window to preview window	N	10
RL_WTOP	Top position of preview window in pixels	N	4
RL_WWIDTH	Width of preview window in pixels	N	4

Figure 2.11 Control Parameters for Runtime Report Preview

The following sections explain the purpose and valid entries for each parameter.

RI_WBORDER

This parameter enables you to control the presence and type of border for the Runtime preview window. You can enter one of the following numeric values:

- 0** results in a preview window with no border.
- 1** results in a fixed-size preview window with a single-line border.
- 2** results in a variable-size preview window with a single-line border.
- 3** results in a fixed-size preview window with a double-line border.

RI_WCTRL

Use this parameter to specify whether the preview window will have a system control box (for switching to other applications or for closing the preview window) in the caption bar. If this parameter contains a logical **True** value, the window will have a control box; if **False**, the window will not have a control box.

RI_WHEIGHT

This parameter controls the height of the preview window. Enter the height value in screen pixels.

RI_WLEFT

This parameter controls where the left edge of the preview window will be anchored. Enter the position in screen pixels.

RI_WMAX

Use this parameter to control whether the preview window will have a maximize control in the caption bar at runtime so that a user can run a report full-screen. If this parameter contains a logical **True** value, the window will have a maximize control; if **False**, the window will not have a maximize control.

RI_WMIN

Use this parameter to control whether the preview window will have a minimize control in the caption bar at runtime. If this parameter contains a logical **True** value, the window will have a minimize control; if **False**, the window will not have a minimize control.

RI_WPARENT

If you want the R&R preview window to be the child of another window, use the RI_WPARENT parameter to specify the window handle of the parent window.

RI_WTOP

This parameter controls where the top of the preview window will be anchored. Enter the position in screen pixels.

RI_WWIDTH

This parameter controls the width of the preview window. Enter the width value in screen pixels.

Understanding the Runtime Status File

While the Runtime is executing, it writes status information into a file (either a table or a text file) and, if the RI_DISPERR flag is T, may display error message boxes. For an explanation of the RI_DISPERR flag, see the **Using Control Tables and Files** section.

The status file created by the Runtime is called RRUNOUT.DBF (if you are using a control table) or RRUNOUT.OUT (if you are using a text control file), unless you use the /O switch with the Runtime command to specify a different file name. The Runtime creates the status file in the current directory, overwriting any existing status file of the same name.

After calling the Runtime, you should check the status file for information about Runtime processing. If the Runtime encountered an error, the file will contain an error message that explains why a report was canceled, as well as an error code that lets you determine the type of error. The status file also contains the number of reports and pages output, so a report can be restarted where it left off.

To avoid being confused by multiple status files, delete existing status files before calling the Runtime. If you are using the Runtime on a network, use /O to specify a unique status file for each user.

Status File Fields

Figure 2.12 lists the fields in the Runtime status file. Each field name has the prefix RO_ to indicate that it provides information about **Runtime Output**.

Field Name	Contents	Data Type	Width
RO_ECODE	Error code	Character	1
RO_EMSG	Error message	Character	80
RO_REPORTS	Number of reports completed	Numeric	5
RO_PAGES	Number of pages completed	Numeric	9
RO_RIRECNO	Last report processed	Numeric	5

Figure 2.12 Fields in Runtime Status File

RO_ECODE

The error code field contains one of the following characters:

- N** – the Runtime completed without error; RO_EMSG is blank.
- C** – the user selected Cancel to cancel a report; the RO_EMSG message is “Report canceled.”
- J** – the Runtime command or the control table contains a syntax error; see the message in the RO_EMSG field and/or the number in the RO_RIRECNO field.
- R** – the report definition or the value in a control table field caused an error in the report; see the RO_EMSG field.

RO_EMSG

The error message field is blank if Runtime processing completed without an error. If processing was canceled for any reason, this field contains the error message. If RO_ECODE contains **C**, the message is “Report canceled,” meaning the user canceled a report. If RO_ECODE contains **R**, the message is the same as the one that displays when you attempt to output the report from within R&R.

If RO_ECODE contains **J**, there is an error in the Runtime command or in the control table. If the error is in the control table, the record number of the erroneous record is returned in the RO_RIRECNO field. Use this record number along with the error message to find and correct the error.

RO_REPORTS

This field contains the number of reports that completed successfully. For example, if you call the Runtime to print three reports and the printer jams during the second report, this field contains the number 1, indicating that one report completed successfully. Use this number to determine which report did not complete and to restart it.

RO_PAGES

This entry contains the number of the last page completed in the report (or in the most recently processed report, if the Runtime command specifies multiple reports). If a report terminated due to an error, the entry contains the number of the last page completed before the error occurred. Use this number to restart a canceled report at the page where the error occurred.

For example, if you are printing pages 10 through 20 of a report and the printer jams on page 15, this entry will contain 14 (the number of the last page that printed successfully). If RO_PAGES contains 14, you can restart the report at page 15 by entering 15 in RI_BEGPAGE and 20 in RI_ENDPAGE.

Note that the Runtime does not update the RO_PAGES field after each page unless the value of RI_CHKTIME in the control table is **P**. See the section in this chapter entitled **Creating and Using Control Tables and Files** for a description of RI_CHKTIME.

RO_RIRECNO

This field contains the control table record number of the last report processed, whether or not it completed successfully. You can use this number together with the error message in RO_EMSG to find and correct an error in the control table. This number is also useful for restarting a report that was part of a multi-report job, since it is the control table record number of the last report that was processed.

Application Calls to the Runtime Executable

This section provides examples illustrating how you can incorporate calls to the Runtime in C, Visual Basic, FoxPro for Windows, and PowerBuilder.

Note that these examples are provided only to demonstrate the syntax for calls to the R&R Runtime for each language.

If your application includes logic to check the Runtime status file, keep in mind that up-to-date information from that file will not be available until the Runtime reports complete. To avoid confusion between old and new status files, delete any existing status files before calling the Runtime.

Calling the Runtime from C

The Windows API provides a function named WinExec for executing programs. To call the Runtime from a Windows C program, you could include a function such as the one illustrated in Figure 2.13.

```
BOOL RunThisReport (LPSTR lpRunin, int iReport, LPSTR lpRptlib)
{
    // Run a single report.
    // Input:
    //   lpRunin           pointer to Runtime control table name
    //   iReport            control table record number of report
    //   lpRptlib           pointer to default library directory name
    // Output: FALSE if Windows could not execute the Runtime program

    char szBuffer [128];
    UINT error;

    wsprintf(szBuffer,"rrwrn %s %d /R%s",lpRunin,iReport,lpRptlib);
    if ((error=WinExec((LPSTR)szBuffer, SW_SHOW)) < 32)
    {
        LoadString(hAppInst, EXE_ERR+error, szBuffer, sizeof(szBuffer));
        MessageBox(hAppWnd, szBuffer, szAppname, MB_ICONSTOP);
        return FALSE;
    }
    return TRUE;
}
```

Figure 2.13 Calling the Runtime from a C Program

In this example, the call to the Runtime includes the /R switch to specify a default report library directory. You can include any combination of the command switches explained in the section of this chapter entitled **Command Switches**.

Note that the second parameter supplied to WinExec, SW_SHOW, is ignored if you have included control table values to govern the Runtime display.

Calling the Runtime from Visual Basic

Visual Basic provides a function named Shell that takes two arguments: a command-line string and a Windows display style. The Runtime ignores the second argument if display characteristics are specified by Runtime control table values.

Figure 2.14 illustrates a subroutine that could be used to call the Runtime from a Visual Basic application.

```
Sub Command1_Click ()  
    cmd$ = "c:\rrw\rrwrun.exe c:\rrw\rrsample\rrwrunin 1 /WN"  
    i% = Shell(cmd$, 1)  
End Sub
```

Figure 2.14 Sample Visual Basic Subroutine Calling the Runtime

In this example, the Runtime is executed using the Runtime control table in C:\RRW\RRSAMPLE. The /WN switch specifies that database users will *not* have write access to the database files used by the Runtime reports while those reports are being generated. Note that you can use any combination of Runtime command switches.

Calling the Runtime from Windows FoxPro

You can execute the Runtime from FoxPro for Windows using the FoxPro RUN command. For example, you could include the following line to call the Runtime from a FoxPro for Windows application:

```
RUN /N RWRUN ACCOUNTS 1 /RC:\DEPT99
```

In this example, the Runtime is executed using a control table named “accounts.” The /R switch specifies a default library directory of c:\dept99. The /N switch is used in FoxPro to execute another Windows-based application.

Calling the Runtime from PowerBuilder

You can execute the Runtime from a PowerBuilder script using the SetProfileString command to change any of the parameters in the control file and the RUN command to execute the Runtime.

Figure 2.15 illustrates a script that could be used to call the Runtime from a PowerBuilder application.

```
SetProfileString("D:\RRWRUN.IN","rrwrn","ri_copies","2")
run("RRWRUN.EXE /Td:\rrwrn.in /Od:\rrwrn.out")
```

Figure 2.15 Sample PowerBuilder Script Calling the Runtime

Parameter Passing

You can control some features of the layout and content of reports at runtime by prompting users to enter values for parameters, then passing the values to reports. Typically, you prompt the user for a text string or other data item that is not stored in the database. For example, you might prompt the user for his or her name and use the name in a “Report Author” field in the page footer or title.

You can also use parameter passing to control report processing at runtime. You can pass parameters to a report in two ways:

- Define special parameters in the Runtime control table or file. At runtime, prompt the user to enter values that are stored in these parameters. Pass the values to your report using a calculated field whose expression includes the RIPARAM() function.
- Create your own menus and prompts and store user entries and selections in a special parameter table. When you create the report, link the parameter table to the report’s master table and use values from the parameter table in the report.

Passing Control Parameter Values

Follow these general steps to pass values to reports using control table or control file parameters.

1. In the Runtime control table or file, define parameters for values you want to pass to the report.
2. Prompt the user to enter a value for the parameter in one of two ways:
 - Create your own menus or prompts within your application.
 - Enter a question mark as the value of the control table parameter.
3. Incorporate the user’s entry into the report using the RIPARAM() function in a calculated field expression.

The following sections describe each step in detail.

Defining Control Parameters

In addition to the predefined parameters listed in Figures 2.3 and 2.11, your control table or file can include parameters you define. A user-defined parameter can have any name and can be up to 512 characters wide. You can define as many parameters as you need for your application. The control table or file need not include any user-defined parameters. If any are present, the table or file need not contain values for all of them.

Prompting for User Input

You can get user input in two ways:

- Supply a menu or prompt in your application that leads the user to supply a value. Store this value in the user-defined parameter in the Runtime control table or file.
- Enter a question mark (?) value for any user-defined parameter. Whenever a user-defined parameter contains a question mark, the user will be prompted to enter a value.

Using the Question Mark Parameter Value

The simplest way to get user input for reports is to use a question mark (?) as the value for a user-defined parameter. Optionally, the value can also include the text you want to appear as a prompt.

For example, if you want to prompt the user for his or her name, you might create an AUTHOR parameter in your control table or file and give it the value “?Enter your name:”. At runtime, the user will see the dialog box shown in Figure 2.16.

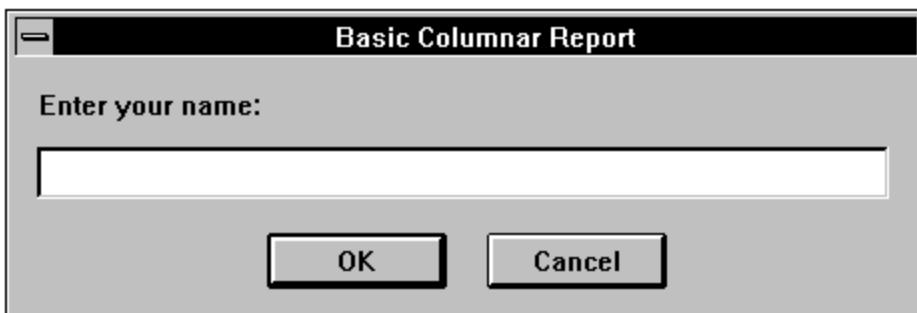


Figure 2.16 Runtime Dialog Box with Prompt

The size and shape of this dialog box is the same for all user-defined parameters. The title bar contains the value of RI_WTITLE. If RI_WTITLE is not specified, the Runtime uses the report name. For control tables, the length of the field in the control table determines the maximum length of the user's entry. For example, if the length in the control table is 30 characters, the Runtime will not allow the user to enter more than 30 characters. If the user selects Cancel, the report will not run and the Runtime will write the "Canceled" message to the status file.

If your control parameter contains a question mark only and no text string, the Runtime displays the dialog box shown in Figure 2.16 with the prompt "Enter value for (FIELD NAME)", as in "Enter value for AUTHOR."

Incorporating User-Supplied Values in Reports

Once the user has entered or selected a value that is stored in the Runtime control table or file, you pass the value to your report using the RIPARAM() function in a calculated field expression. RIPARAM() takes a control parameter name as its argument and returns the parameter's value as a string.

For example, in a general ledger application, you might create a control parameter CONAME for the company name, then prompt the user to enter a company name. To use the company name on the report, create a calculated field in R&R whose expression is:

```
RIPARAM("CONAME")
```

You can place the calculated field wherever you want the company name to appear on the report.

Although this example uses an RIPARAM() calculated field to provide user input as text in the report, you can use such fields to perform many different functions in a report. For example, you might prompt the user for a value for a DISCOUNT field. In the calculated field on the report, you can convert the user-entered character data to numeric using a calculated field expression such as:

```
ORDERTOT * VAL(RIPARAM("DISCOUNT"))
```

Using a Parameter Table

You can also pass parameters to a report by storing user-supplied values in a separate table called a *parameter table* and relating this table to the report's master table.

To pass parameters to a report using a parameter table, follow these steps:

1. Decide on the parameters you need and create a parameter table using your Xbase software.
2. In interactive R&R, create a new report or retrieve and modify an existing report.
3. Create a calculated field that you will use to link the report's master table to the parameter table.
4. Relate the master table to the parameter table using the R&R calculated field as the linking field from the master table.
5. Use the fields from the parameter table in your report and save the report.
6. Create an application program that will get the information from the user, store it in the parameter table, and call the Runtime to generate the report.

The following sections describe steps 1 through 4 in more detail and provide an example of using a parameter table in an application.

Creating the Parameter Table

A parameter table can contain as many parameters as you need for your application. For example, you may want the user to supply a date, a range of dates, an account number, a category name, a list of items, or a logical true/false flag. Include one field for each parameter. You can use any data type. Assign a field width sufficient for your purpose.

Most likely, your parameter table will have a single record. However, if you want to create one parameter table for use with multiple reports, the table should have a record for each report. For each report, append one blank record to contain the parameters.

Creating the Calculated Linking Field

In interactive R&R, use Calculations ⇒ Calculated Field to create a calculated linking field whose expression is a constant equal to the record number or index key value for the appropriate record in the parameter table. If your parameter table has a single record, you can simply use the number 1 as the calculated linking field value.

If your parameter table has multiple records, you might use the key field value as the calculated linking field value. For example, if the parameter

table's key field is the report name, the calculated field's expression is the report name that identifies the appropriate record, as in "INVOICE1."

Relating the Master Table and the Parameter Table

Use Database ⇒ Relations to relate the master table and the parameter table. Use the calculated field as the linking field from the master table and the record number or key field as the linking field from the parameter table.