Designing Report Builder Parameter Passing for Deployment with Report Engine in OpenEdge 10

Authors: Jim Lundy, Salvador Viñals
Edition 1, October 2003

Contents

Parameter Passing with Report Engine

Progress Software has announced in its “Statement of Direction” for Business Intelligence, and at events such as Exchange 2003, that it will no longer include the Report Builder development tool after the last release of Progress® Version 9.1. However, the Report Engine deployment environment (runtime) will continue to be available in OpenEdge™ 10 products, on Windows 32-bit platforms only.

There have been several inquiries asking if and how parameter-passing features of existing Report Builder reports will work in the runtime-only Report Engine environment of OpenEdge 10. The purpose of this white paper is to provide direction and best-practices on how to structure Report Builder reports in Progress Version 9 so that they will be able to run properly in an OpenEdge 10 Report Engine environment.

Parameters and Passing Methods

The Report Engine runtime has the ability to accept parameters supplied at runtime. This capability will continue to be available in the Report Engine runtime that will be supplied in OpenEdge 10.

There are two types of parameters: pre-defined and user-defined[1]:

Predefined parameters are parameters defined by Report Engine to identify the report you want to run and to control frequently changed report features, such as filters and print information, allowing you to customize your reports. There is also an output parameter that provides status information.

User-defined parameters are character string arguments that you specify in the RB-OTHER-PARAMETERS field. Regardless of which interface you use, you can specify multiple user-defined parameters or arguments, separating the arguments with the linefeed character. The Report Engine expects to find them all in a single RB-OTHER-PARAMETERS value. For example (where “~n” represents the linefeed character when used in Progress 4GL):

"City = Boston~nState = MA~nQuarter = Third-Quarter"

There are three methods that can be used for passing runtime parameters to Report Builder reports:

- **Table Interface—**A database table can be created and loaded with the parameters. At runtime, the parameters can be read from the table and the report(s) processed in a batch-like manner. This solution can be used with both pre-defined and user-defined parameters and is ideal for cases where there are many reports to run or manual entry of parameters would be complicated or time-consuming.[2]
- **Runtime Passing—**The Report Engine runtime has the ability to accept input parameters at report runtime. This solution can be used with both pre-defined and runtime parameters and is ideal for cases where parameters need to be passed programmatically.
- **Runtime Prompting—**The Report Builder report can be written so that it will prompt the user for parameter input at runtime. This solution can be used with both pre-defined and runtime parameters and is ideal for cases where parameter values are unknown prior to runtime or are determined by the user. The user must be using the machine where Report Engine is running.

---

[1] Refer to the Progress Report Builder Deployment Guide, Chapter 3, “Report Engine Parameters” for additional details on parameter types.

[2] Refer to the Progress Report Builder Deployment Guide, Chapter 4, “Report Engine Table Interface”, for additional details.

Examples

Table Interface

The Report Engine table interface stores the report parameters in a database table called the Report Engine table, then uses the parameters to generate one or more reports. The fields in the Report Engine table that store the report parameters are called Report Engine fields. These fields contain values for the required and optional report parameters. A Report Engine record is a record that contains the Report Engine fields. Being able to store report information in the Report Engine table allows you to run batches of reports without running the Progress 4GL. Another advantage of using the table interface instead of the PRINTRB or PRNTRB2 interface is that the table interface allows you to invoke Report Engine directly from the command line or a Windows icon.

Follow these steps to create the Report Engine table using the rbreport.df file, and then invoke Report Engine using it:

1. Create a database called Runtable containing the Report Engine table. You can use the database definition file (%DLC%\bin\rbreport.df) that contains loadable definitions for the RBREPORT table. (Note that you can give the database any name you choose, and it can contain other tables, such as the tables that contain your report data.)
2. Create a Windows icon with the following command on the command line:

```
wproserv -db dbname
```

Note: the dbname contains the full pathname of the Runtable database.

3. Double-click the icon to start the database server. The icon becomes minimized.
4. Start Progress and connect to the Runtable database in multi-user mode.
5. Start the Procedure Editor.
6. Run the rbstart1.p procedure.

Here is the code for the rbstart1.p procedure:

rbstart1.p

```
DO TRANSACTION:
 /* 1 */
CREATE RBREPORT.
/* 2 */
ASSIGN
RBREPORT.RB-REPORT-LIBRARY = "c:\dlc\src\aderb\rbsample.prl"
RBREPORT.RB-REPORT-NAME = "Customer Discount"
RBREPORT.RB-PRINT-DESTINATION = "D"
RBREPORT.RB-DISPLAY-STATUS = yes
RBREPORT.RB-DISPLAY-ERRORS = yes.
RELEASE RBREPORT.
END.
/* 3 */
RUN aderb\_prore(false, "-db Runtable -S servername -H hostname -N networktype -rbdel").
```

The commented numbers correspond to the following step-by-step descriptions:

1. Create an RBREPORT record.
2. Enter the values for five of the record fields. Report Engine uses the default values for the fields for which you do not specify a value.

3. Call the _prore.p procedure to invoke Report Engine and specify the NO-WAIT-value and the database connection information. By specifying "false" as the NO-WAIT-value, you instruct Progress to wait until Report Engine completes processing before continuing with the application. Finally, the Report Delete (-rbdel) parameter deletes the record from the Report Engine table when the procedure is done.

Runtime Passing

You can prompt a user for other report information using a user-defined parameter.
In the "User-defined Parameters" section, the example shows how to print the report author's name at the top of the report. To prompt the user to enter his or her name, follow these steps:
1. In the Report Builder, create a calculated field named MYNAME with the following expression:

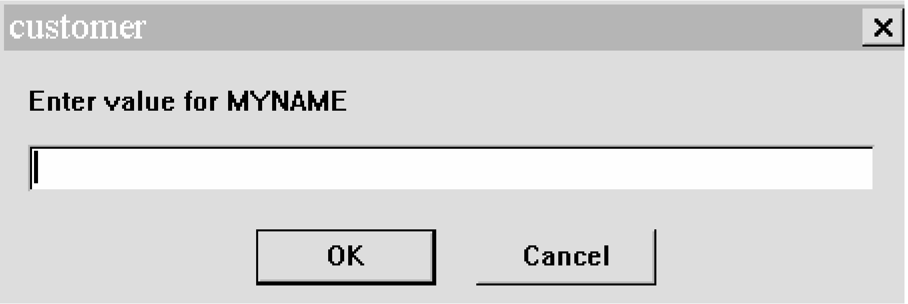
```
RUNTIME-PARAMETER("MYNAME")
```

2. Place MYNAME in the title band line.

3. When you run the Report Engine, you can use the Report Engine prompt to prompt the user to enter his or her name using the following code in the RB-OTHER-PARAMETERS parameter:

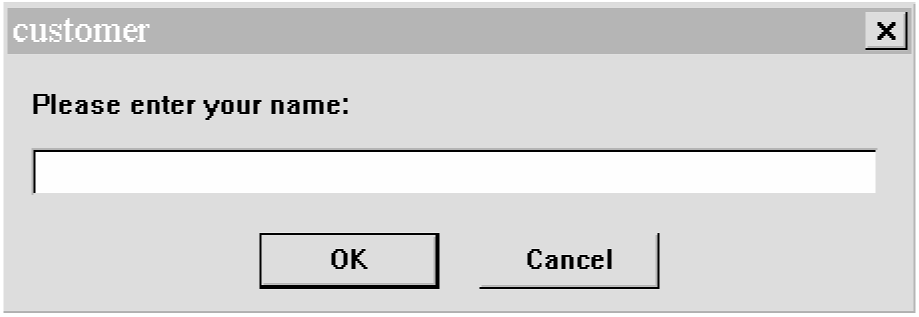
```
"MYNAME = ?"
```

This causes the Report Engine Parameter dialog box shown below to appear:



Enter value for MYNAME

Customize the prompt using the following code:
"MYNAME = ?Please enter your name:"
The Report Engine Parameters dialog box will appear as shown below:



Please enter your name:

When you prompt users for input, the Report Engine Parameter dialog box title bar contains the value of the RB-WINDOW-TITLE parameter. If RB-WINDOW-TITLE is empty or UNKNOWN, Report Engine uses the report name as the dialog box title.

You cannot modify the dimensions of the dialog box to accommodate the anticipated user input. The dialog box always remains the same size and users can enter up to 512 characters. If the user chooses the Cancel button, Report Engine does not run the report and writes the "Cancelled" message to the report status file or the RB-STATUS field, depending on the interface and command-line parameters you use.

Runtime Prompting

The rbfilt2.p sample procedure runs the Customer List report in the rbsample.prl report library and prompts the user for minimum and maximum values for the filter condition:

rbfilt2.p

```
DEF VAR high-value AS INTEGER INITIAL 0.
DEF VAR low-value AS INTEGER INITIAL 0.
DEF VAR rb-filter-value AS CHARACTER INITIAL "".
/* 1 */
FORM" Enter Low  Value for CUSTOMER NUMBER: " low-value at 20  SKIP
"Enter High Value for CUSTOMER NUMBER: " high-value at 20
WITH FRAME TEST-FRAME CENTERED NO-LABELS.
/* 2 */
UPDATE low-value high-value WITH FRAME TEST-FRAME.
HIDE FRAME TEST-FRAME.
/* 3 */
rb-filter-value = "Customer.Cust-num >= " + STRING(low-value) +
" AND Customer.Cust-num <= " + STRING(high-value).
/* 4 */
DO TRANSACTION:
CREATE RBREPORT.
ASSIGN
RBREPORT.RB-REPORT-LIBRARY = "c:\dlc\src\aderb\rbsample.prl"
RBREPORT.RB-REPORT-NAME = "Customer List"
BREPORT.RB-PRINT-DESTINATION = "D"
RBREPORT.RB-INCLUDE-RECORDS  = "O"
RBREPORT.RB-FILTER  = rb-filter-value
RBREPORT.RB-DISPLAY-ERRORS = yes
RBREPORT.RB-DISPLAY-STATUS = yes.
RELEASE RBREPORT.
END.
RUN  aderb\_prore.p(false, "-db Runtable  -S servername -H hostname -N networktype -rbdel").
```

The commented numbers correspond to the following step-by-step descriptions:

1. Define a form for the prompt.

2. Prompt the user for filter information.

3. Assign the filter override condition to rb-filter-value.

4. Run the Customer List report with the filter override on the report.

Related Topics

The Report Engine PRINTRB and PRNTRB2 interfaces allow you to invoke the Report Engine from the Progress 4GL using parameters instead of tables and fields to specify report parameters as shown in the examples above. Refer to the Progress Report Builder Deployment Guide, Chapter 5, “Report Engine PRINTRB and PRINTRB2 Interfaces”, for additional details.

Corporate and North American Headquarters
Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA Tel: 781 280 4000
Fax: 781 280 4095

Europe/Middle East/Africa Headquarters
Progress Software Europe B.V. Schorpioenstraat 67 3067 GG Rotterdam, The
Netherlands Tel: 31 10 286 5700 Fax: 31 10 286 5777

Latin American Headquarters
Progress Software Corporation, 2255 Glades Road, One Boca Place, Suite 300 E, Boca
Raton, FL 33431 USA Tel: 561 998 2244 Fax: 561 998 1573

Asia/Pacific Headquarters
Progress Software Pty. Ltd., 1911 Malvern Road, Malvern East, 3145, Australia Tel: 61 39
885 0544 Fax: 61 39 885 9473

PROGRESS
SOFTWARE

www.progress.com