

DataLink Viewer 2011

View, Print, and Export Crystal Reports

www.MilletSoftware.com

Version 6.2

May 2012

By

Millet Software

5275 Rome Ct.

Erie, PA 16509-3951

ido@MilletSoftware.com

(814) 825-6009

Disclaimer: This software is provided “as-is” by Millet Software without assuming any responsibility for harm to computer systems, software, or data with which these files are used.

INTRODUCTION	6
MAIN BENEFITS:.....	6
INSTALL / REMOVE	8
TYPICAL USE SCENARIOS	9
SELECT REPORT WINDOW	9
Right-Click Report Row Menu	10
PREVIEW REPORT WINDOW:	11
Function Keys and Report Grid Options	13
Selecting a Report to View.....	14
COPYING TEXT CONTENT FROM THE REPORT	14
Text Copy (Ctrl-C).....	14
Tooltip Copy (Ctrl-Shift-C)	14
PARAMETER FUNCTIONALITY	15
REMEMBER PARAMETER VALUES FROM A PREVIOUS SESSION	15
SELECTIVE PARAMETER REFRESH	15
SAVE AND REUSE NAMED PARAMETER SETS	16
CLICK TO CHANGE A PARAMETER VALUE.....	18
Toggling Between 2 Parameter Values	18
Special Date Parameter Dialog	18
Input Box Dialog.....	18
ListBox Dialog.....	19
Changing a Parameter by Clicking on a Related Formula	19
LOAD INI VALUES INTO PARAMETERS	20
SECURING REPORTS AGAINST UNAUTHORIZED USE	20
LOAD MACHINE/COMPANY/REPORT INFORMATION INTO PARAMETERS	21
DLV_Machine_Name	21
DLV_Registered_Company	21
DLV_HD_Serial_N.....	21
DLV_Rpt_Path.....	21
CONTROL DATA ACCESS ACCORDING TO PC LOGIN.....	22
DYNAMIC & CASCADING PARAMETERS	23
Respond to Single-Value Live Prompts	23
Link back in a Hierarchy of Live Prompts	24
Respond to Multi-Value Live Prompts	25
Manage Display & Data Values for Live Parameters	26
View the Report	27
DYNAMIC AND CASCADING PARAMETERS (DEVELOPER NOTES)	28
Naming Parameters	28
Designing Live Prompt Reports	28
Implementing a “Select All” Option	29
Sharing Values Across Cascading Parameters	30
Requiring a Value for a Dynamic Parameter	31
Forcing Users to Select from the List of Values (No Direct Edit)	31

AUTO-REFRESH REPORTS.....	32
AUTO_REFRESH COMMAND LINE ARGUMENT	33
AUTO_PAGE_AND_REFRESH COMMAND LINE ARGUMENT.....	33
VIEWMODE COMMAND LINE ARGUMENT (REMOVE TOOLBAR/STATUS BAR).....	33
CYCLING THROUGH SEVERAL AUTO-REFRESHED REPORTS.....	34
CLICK TO SET FORMULA VALUE	35
CLICK COLUMN HEADERS TO RE-SORT THE REPORT.....	35
IN-PLACE DRILL-DOWN	36
CONVERTING SECTION DOUBLE-CLICK TO IN-PLACE DRILL-DOWN	38
DYNAMIC GROUPING (GROUP SWAP EXPERT)	39
OPTIONAL TECHNIQUE FOR GAINING FORMULA ACCESS TO THE DYNAMIC GROUPING:.....	40
DATA VISUALIZER.....	41
LAUNCHING REPORTS & COMMAND LINE API.....	44
LAUNCH REPORTS FROM FILE EXPLORER.....	44
LAUNCH REPORTS FROM COMMAND LINES.....	45
EXAMPLES	45
CALL DATA LINK VIEWER FROM ANOTHER APPLICATION.....	46
COMMAND LINE ARGUMENTS FOR PARAMETER VALUES.....	47
Range and Multi-Value Parameters	47
Optional Parameters with No Value.....	47
Null Values	48
Ignoring Saved Parameter Values	48
COMMAND LINE ARGUMENTS FOR TRIGGERING EXPORTING.....	48
COMMAND LINE ARGUMENTS FOR TRIGGERING PRINTING	49
Specifying Number of Copies	49
Scheduling Printing	50
Scheduling Printing for Multiple Reports	50
ARGUMENT FOR DOCUMENTING FILE LOCATIONS AND REPORT PARAMETERS.....	51
LAUNCH A REPORT WHILE VIEWING ANOTHER REPORT	52
CREATE A USER INTERFACE FOR SELECTING AND LAUNCHING REPORTS.....	53
EMBED INPUT FROM THE USER IN THE COMMAND LINE CALL FOR ANOTHER REPORT.....	54
LAUNCH ANOTHER APPLICATION AND PASS PARAMETERS TO IT	55
LAUNCHING REPORTS IN A NEW WINDOW	56
New Window On File Launch and/or On Double Click	56
Right-Click the Grid and Select Preview Report (new Window).....	57
Remove the Preview Tab from the Initial Window.....	57
DATABASE CHOICE FUNCTIONALITY.....	58
SELECT ALTERNATIVE ODBC DATA SOURCES FOR THE SAME REPORT.....	58
RESTRICTING THE LIST OF ODBC DSN CHOICES.....	59

USING COMMAND LINE ARGUMENTS TO SPECIFY THE ODBC DSN	60
OVERRIDING THE DATABASE SPECIFIED IN THE REPORT OR ODBC DSN	60
OVERRIDING THE TABLE LOCATION.....	61
OVERRIDING THE XML FILE LOCATION.....	61
STRIPPING TABLE QUALIFIERS WHEN CONNECTING TO A DIFFERENT DATABASE	62
OVERRIDING THE SERVER IN NATIVE ORACLE CONNECTION	63
SELECTING AN ALTERNATIVE SQL SERVER – OLE DB DATA SOURCE	63
FORCED LOGIN	64
SELECTING FOLDER LOCATION FOR FOXPRO DBF FILES (MASTERBUILDER).....	65
CHANGING FOLDER LOCATION FOR ACCESS/EXCEL/PERVASIVE (DDF) FILES	66
Using Command Line Argument to Avoid Path Prompt	66
INTEGRATED AUTHENTICATION	67
INTEGRATED AUTHENTICATION ("REMEMBER ME")	67
SHARED MACHINE AUTHENTICATION	68
SHARED SECRET PASSWORD WITH WINDOWS USER IDS.....	69
PROTECT REPORT DESIGNS WITH RPZ FILES	70
MAKE_RPZ COMMAND LINE ARGUMENT.....	71
CREATE RPZ FILES WITH EXPIRATION AND LICENSE KEYS.....	72
Make_Rpz2 Command Line Argument	73
MONITORING DATALINK VIEWER USE	75
RECORD PROCESSING TO AN ODBC DATABASE.....	75
Typical Use	75
MS Access Data Structure.....	76
SQL Server Data Structure.....	77
Oracle Data Structure.....	78
“Oracle” mode (Bind Variables).....	79
How to Start Logging?	79
RECORD REPORT USE IN A TEXT LOG FILE	80
SETTINGS & OPTIONS	81
CUSTOMIZING THE GRID LAYOUT	81
CUSTOMIZING THE GRID STYLE	82
DISABLING REPORT PREVIEW BUTTONS.....	83
DISABLING DATALINK VIEWER BUTTONS	84
ADD YOUR COMPANY INFO TO THE ABOUT DIALOG	85
CITRIX AND FILE LOCATION FUNCTIONALITY	86
EXPLICIT ASSIGNMENT OF DEFAULT PRINTER.....	87
FILE LOCATION & REDIRECT LOGIC.....	88
Checking and Navigating to Key File Locations	89
ENFORCING SETTINGS IN A MASTER DATALINK_VIEWER.INI FILE	90

UPDATING DATA LINK_VIEWER.INI VIA A DELTA FILE	91
UPDATE HISTORY	92
VERSION 6.2.2070: ENTERED TESTING MAY 5, 2012.....	92
Known Issues and Limitation.....	95

Introduction

DataLink Viewer 2011 uses the Crystal 2011 runtime components and allows you to run Crystal reports from version 7, 8, 8.5, 9, 10, XI, 2008, and 2011.

While the creation and design changes of Crystal reports (.rpt files) require the full **Crystal Reports** software, you can let other PCs view, print, and export these reports by installing **DataLink Viewer**.

DataLink Viewer provides several useful features such as **command line API** (allowing you to **schedule printing** and **trigger viewing** of reports from your application, task scheduler, batch files, or desktop shortcuts), an intuitive Grid for **organizing and selecting previously opened reports**, reduced login frustrations via **integrated authentication**, choice of **alternative data sources**, **selective parameter refresh**, **dynamic and cascading parameters** even for versions prior to XI, **auto-refresh**, **user-based row-level security**, and more...

Main Benefits:

1. **View, Print, and Export** Crystal reports (7, 8, 8.5, 9, 10, XI, 2008, 2011)
2. **Uses Crystal 2011 runtime components** providing full support for new features such as interactive parameter panel and advanced CrossTabs.
3. **Group Swap Expert allowing** allowing group changes on the fly
4. **Data Visualizer** adding OLAP and advanced charting capabilities
5. **Intuitive Customizable Grid Interface** to classify, organizing, and launch Reports.
6. **Dynamic & Cascading Parameters** (even for pre-XI reports):
 - **Select Live Parameter Values** from linked Crystal reports that act as dynamic pick list data sources or default to using Crystal's static parameters.
 - **Restrict Live Parameter Values** based on choices in a prior live parameter.
 - **Remember Values last used for each Live Prompt** allowing the user to accept or replace those values.
7. **Command Line Interface:** for launching reports from any other program and even from within other Crystal reports.

8. **Protect & Hide Report Designs** by converting your rpt files into rpz files. Your users can run the resulting rpz files in DataLink Viewer, but cannot view or modify them in Crystal. To completely protect report designs, DataLink Viewer blocks exporting of rpz files to rpt or report definition files.
9. **Filter Data based on User Login.**
10. **Export & Print Reports Directly** (without previewing)
11. **Remember Last Export Format and File Name** for Each Report.
12. **Auto-Refresh Reports**
13. **React to User Actions in Useful Ways (requires special formulas):**
 - **In-Place Drill Down** (www.milletsoftware.com/Download/DLV_InPlace_DrillDown.wmv)
 - **Click to Group/Sort** (http://www.milletsoftware.com/Download/DLV_Sort_By_Click.wmv)
 - **Click to set a formula value**
 - **Click to prompt for and pass a value to another process**
14. **Compatibility with document management systems**
such as Documentum.
15. **Selective Parameter Refresh:** When refreshing a report, **users can select which parameters they wish to change.** This avoids tedious re-entering of values for the other parameters. The parameter refresh choices are stored for each report and can be easily reused or changed at a later session.
16. **Integrated Authentication:** use Windows User Login and Machine ID to remove the need to repeatedly authenticate to data sources.
17. **Reduced Memory Requirements** compared to Crystal and other viewers.
18. **Select Different Data Sources** for the same Report.

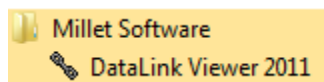
Install / Remove

The exe file you downloaded self-extracts to 2 files: a setup.exe and an msi file. First, it will prompt you to provide a password:



It automatically triggers the setup.exe to check for and download/install any missing prerequisites such as the .NET framework and the Crystal 2011 runtime components. It then triggers the msi install of DataLink Viewer itself.

DataLink Viewer 2011 is installed under the “Program Files\Millet Software.



Avoid Installation on a Crystal Enterprise Machine:

Due to the risk of rare runtime component conflicts, you should avoid installing the software on a machine that also runs Crystal Enterprise.

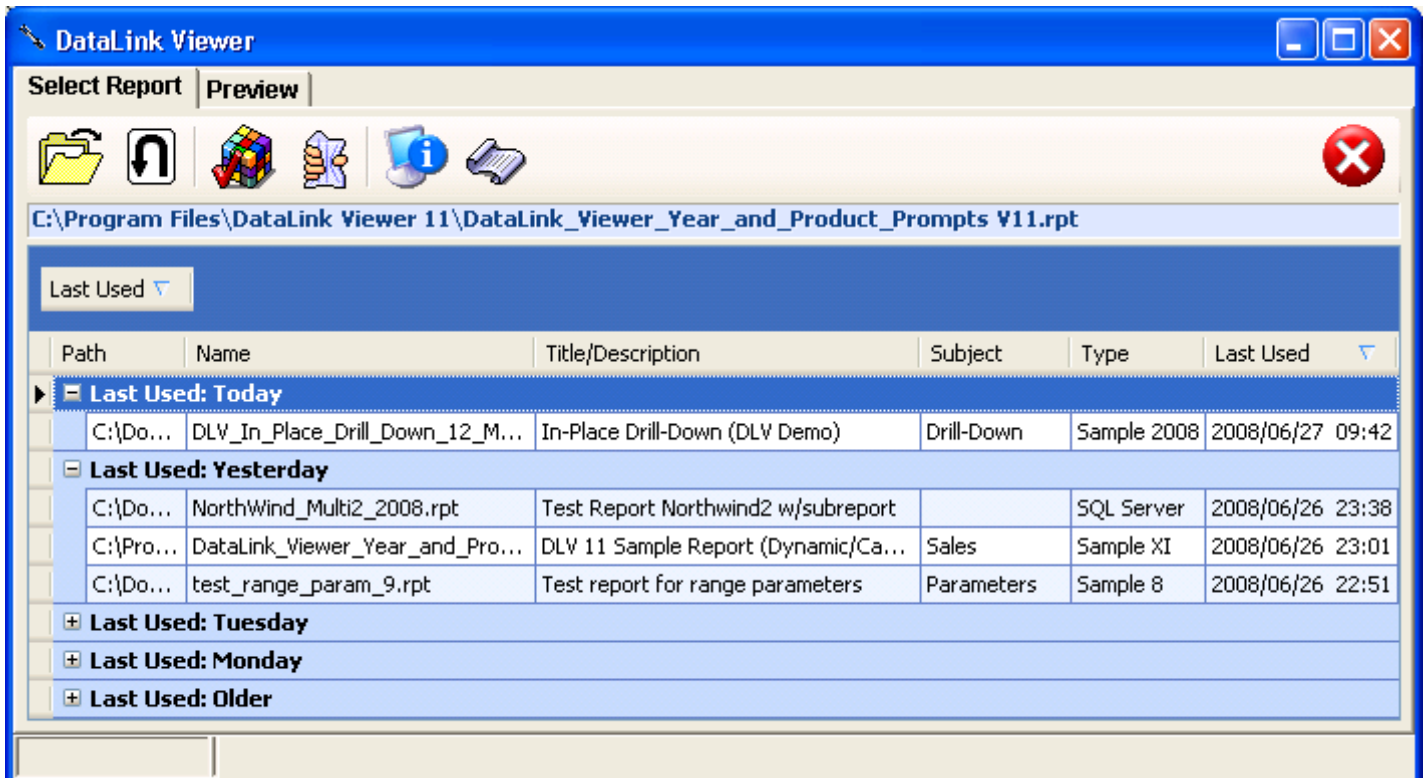
Typical Use Scenarios


In a typical scenario, you would select a report to view by double-clicking it. After prompting you for parameter values, the viewer would display the report with all the preview functionality available within Crystal (drilldown, tree view, export, zoom, print).

The report can also be invoked from a desktop shortcut, a batch file, or another program using command line options.

Select Report Window

After starting *DataLink Viewer*, you would see a screen similar to this:



Use the  button to browse for and open a report for the first time. Previously selected reports are listed in a grid and can be launched by **double-clicking** or by **Right-Clicking** or by selecting and clicking the Preview Tab.




Use the  button to reload a report (if changed and saved in Crystal).




Use the  button to access a dialog for setting various **options**. These options are maintained in the file **DataLink_Viewer.ini**




Use the  button to open this **User Manual** inside MS Word.



Use the  button to **compress & encrypt RPT files into RPZ files**. This allows you to protect and hide your reports designs (either as an intellectual property issue or as a tech support issue). See the section on “Protecting Report Designs with **rpz** Files” for more detail on this feature.



Use the  button to access a dialog with **Version (and system) information**. That dialog also has a button that allows you to check for software patches on my web site and install them online. This patching mechanism is very fast since the patches are typically very small (contain only file changes).



Use the  button to **exit** DataLink Viewer.

RIGHT-CLICK REPORT ROW MENU

if you Right-Click a **report row** in the grid, the following popup menu is displayed:



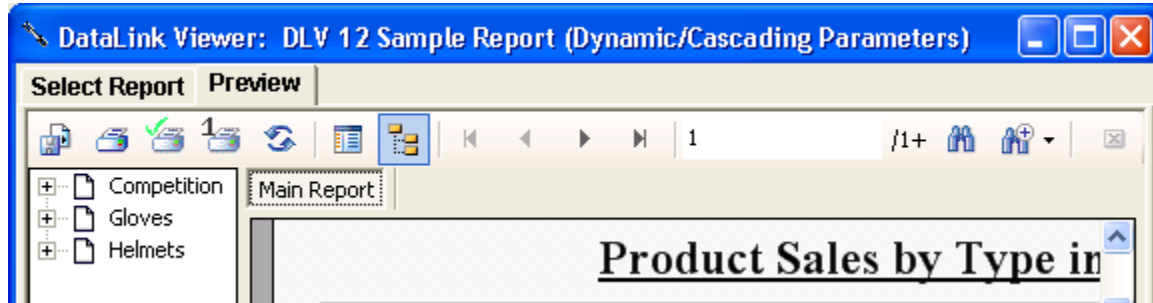
To **delete a report from the grid** (but not from the hard drive) select ‘Delete Row’ from the popup menu or select the report row and hit Ctrl-Delete. The grid information is maintained in a plain text file (**ReportList.txt**).

The other options in the popup menu allow you to **launch a report to a new window, force a login dialog (allowing a choice of a different data source), and printing or exporting the report without previewing it**.

When adding a report to the grid, DataLink Viewer populates the **title** and **subject** columns automatically if it finds that information in the **summary information** for the report. You set that information for the .rpt file in Crystal (under the file, *Summary Information* menu).

Preview Report Window:

The preview window looks similar to the preview window in Crystal except for a few enhancements:



Export Button

The export button “remembers” the last export format and file destination you used for each report. If this is the first time you are exporting a report, the export format and folder will default to those used in the last export.

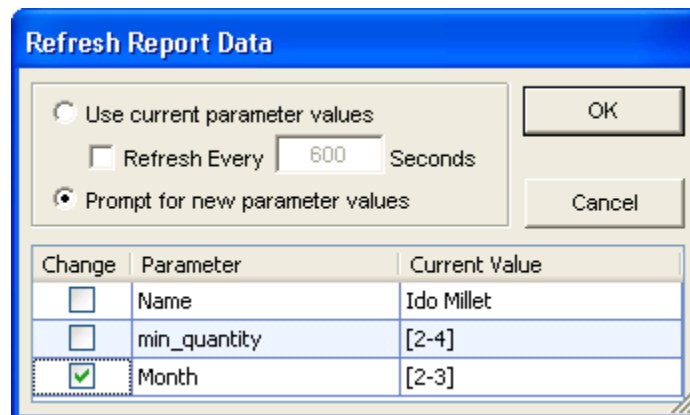
3 Print Buttons

Instead of a single print button, DataLink Viewer gives you:

1. a regular **print** button, which invokes a printer selection and setup dialog
2. a “**Quick Print**” button, which immediately sends the report to the default printer, or
3. a “**Print Current Page**” button, which immediately prints just the current page to the default printer.

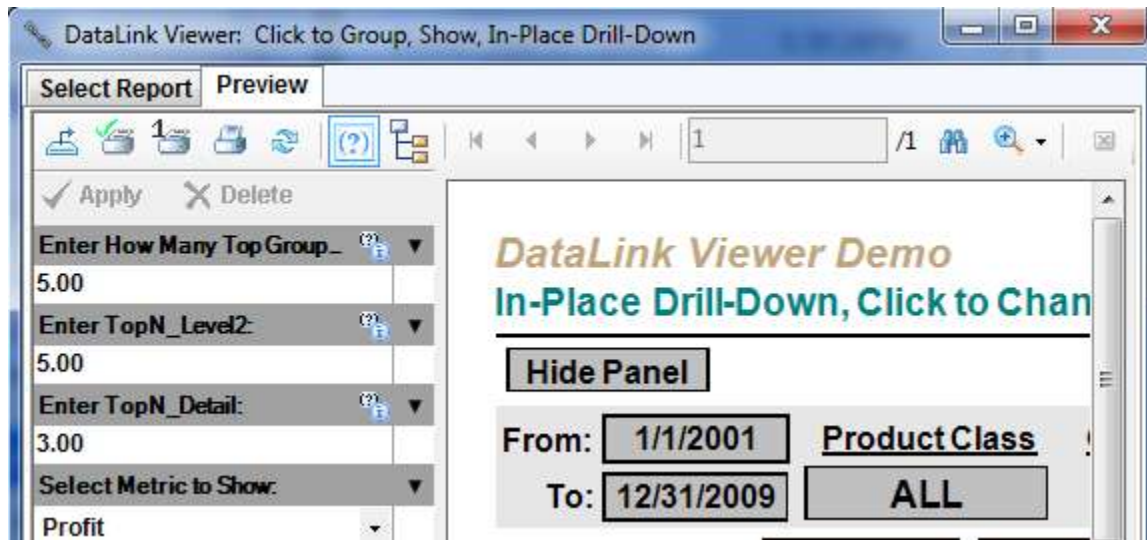
Refresh Button

DataLink Viewer allows you to auto-refresh a report every N seconds and to selectively change the values of only some parameters:



Interactive Parameter Button

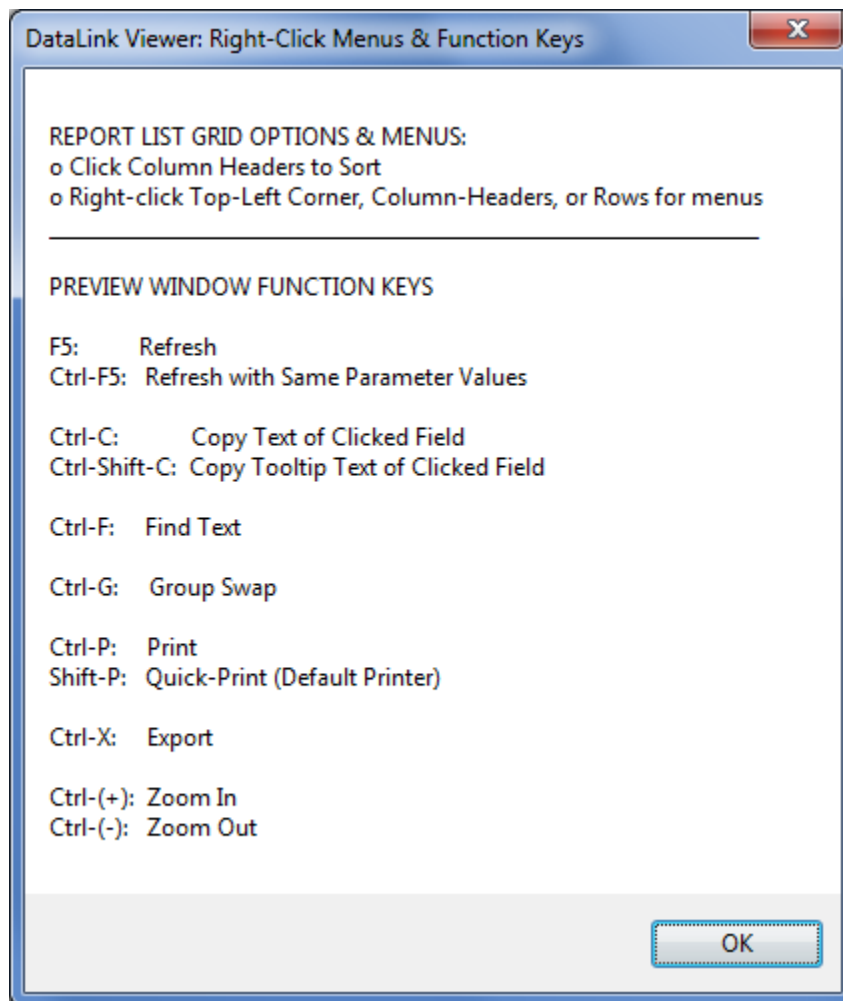
Just like in Crystal, the interactive parameter panel allows you to change parameter values without refreshing the whole report:



The rest of the buttons in the viewer toolbar behave as you may expect

FUNCTION KEYS AND REPORT GRID OPTIONS

If you hit **F1** while in preview mode, the following Help window displays:



SELECTING A REPORT TO VIEW

Let's launch **DataLink_Viewer_Year_and_Product_Prompts V12.rpt** (installed under the **c:\Program Files\DataLink Viewer 12** directory in typical installations). You can do this by

- 1) **Selecting that row and clicking the Preview Tab**, or
- 2) **Right-Clicking that row and selecting Preview**, or
- 3) **Double-clicking the row**.

Note: the sample reports assume that you have the Crystal Reports Extreme Sample Database for that version of Crystal already installed on your PC.

Before displaying the report, DataLink Viewer would prompt you for any report parameters and logon information required by the report. **If the report was saved with data**, the viewer would ask you if you want to refresh the data or use the data saved with the report (in which case you would not be prompted for parameter values).

Copying Text Content from The Report

When previewing a report, you may want to copy the text content from a clicked field or formula.

TEXT COPY (CTRL-C)

After a field/formula is clicked, pressing **Ctrl-C** copies the value of the field as text to the clipboard. You can then paste that text (using Ctrl-V or right-click editing menu to a target of your choice).

TOOLTIP COPY (CTRL-SHIFT-C)

After a field/formula is clicked, pressing **Ctrl-Shift-C** copies the value of the field as text to the clipboard.

Use Scenario: Imagine you have Customer Support Representatives (CSRs) who use Crystal reports to view order history information for calling customers. Frequently, these CSRs need to grab order history information for a specific order within the report, and email that information to the customer. You don't want to display all that history information inside a single field on the report because of formatting and space utilization considerations. But you can accumulate that content using Crystal formula logic into the tooltip expression for a field or a formula. After selecting that field or formula, the CSR can simply hit **Ctrl-Shift-C** to copy the content of the tooltip. They can then paste it to an email message using Ctrl-V.

Parameter Functionality

Remember Parameter Values from a previous Session

[Video Demo](#)

If you refresh or reload the same report, the values you specified last time for each parameter are already selected, allowing you to accept or replace them.

Selective Parameter Refresh

[Video Demo](#)

When refreshing a report that has parameters, a special dialog allows users to select which parameters they wish to change. These choices are stored for each report for reuse/ change at a later session. **Unlinked** subreport parameters participate in this process and are listed as [subreport name] -> Parameter Name.

Change	Parameter	Current Value
<input type="checkbox"/>	Start_Date	11/21/2008 12:00:00 AM
<input type="checkbox"/>	Product_Types	Gloves,Helmets
<input checked="" type="checkbox"/>	Status_Codes	[50-100],10

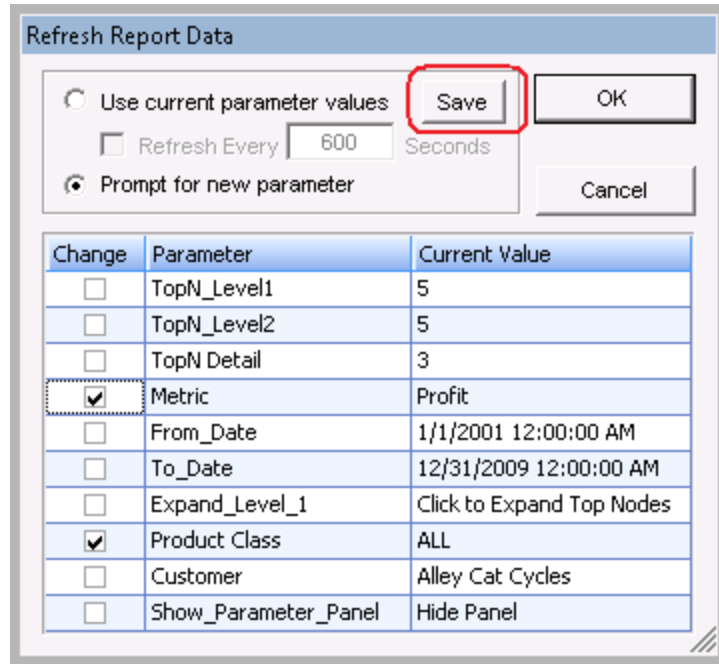
Notes::

- To remove **linked** subreport stored procedure parameters from this dialog, the parameter name must contain “_Linked”.

Save and Reuse Named Parameter Sets

[Video Demo](#)

If you click the refresh button for a report that has at least 8 parameters (Options dialog allows you to change that number), a **Save** button becomes visible in the following dialog:

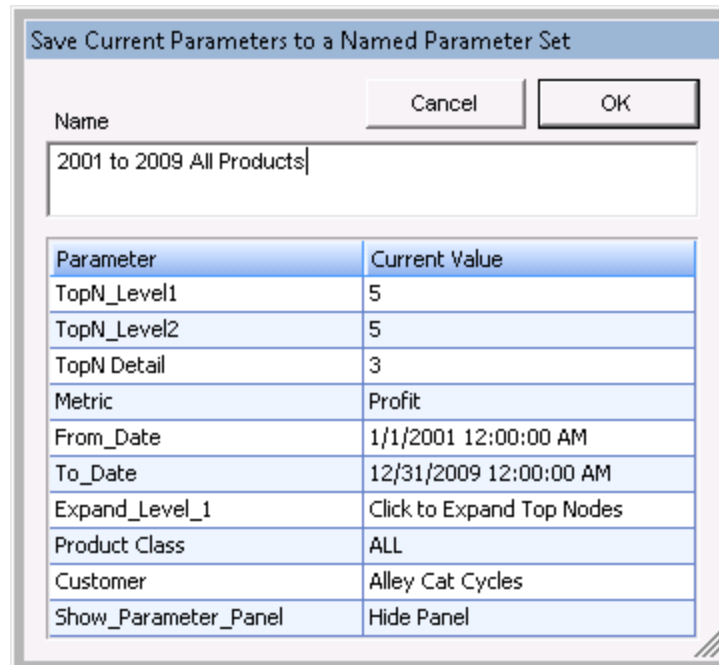


The "Refresh Report Data" dialog box contains the following elements:

- Radio button: Use current parameter values
- Checkbox: Refresh Every Seconds
- Radio button: Prompt for new parameter
- Buttons: Save, OK, Cancel
- Table with columns: Change, Parameter, Current Value

Change	Parameter	Current Value
<input type="checkbox"/>	TopN_Level1	5
<input type="checkbox"/>	TopN_Level2	5
<input type="checkbox"/>	TopN Detail	3
<input checked="" type="checkbox"/>	Metric	Profit
<input type="checkbox"/>	From_Date	1/1/2001 12:00:00 AM
<input type="checkbox"/>	To_Date	12/31/2009 12:00:00 AM
<input type="checkbox"/>	Expand_Level_1	Click to Expand Top Nodes
<input checked="" type="checkbox"/>	Product Class	ALL
<input type="checkbox"/>	Customer	Alley Cat Cycles
<input type="checkbox"/>	Show_Parameter_Panel	Hide Panel

If you click **Save**, the following dialog allows you to save the current parameters value set to DataLink_Viewer.ini under a unique name for this report.

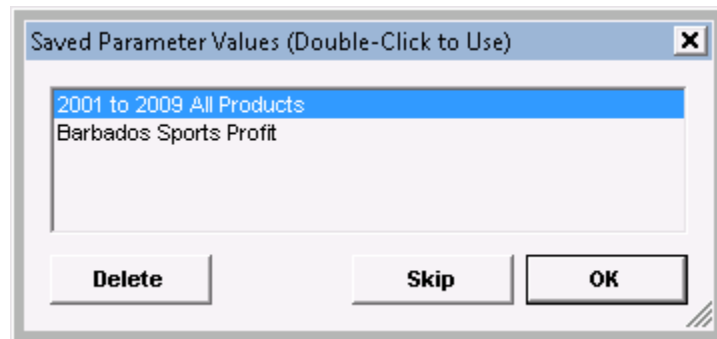


The "Save Current Parameters to a Named Parameter Set" dialog box contains the following elements:

- Text input field: Name
- Buttons: Cancel, OK
- Table with columns: Parameter, Current Value

Parameter	Current Value
TopN_Level1	5
TopN_Level2	5
TopN Detail	3
Metric	Profit
From_Date	1/1/2001 12:00:00 AM
To_Date	12/31/2009 12:00:00 AM
Expand_Level_1	Click to Expand Top Nodes
Product Class	ALL
Customer	Alley Cat Cycles
Show_Parameter_Panel	Hide Panel

The next time you load this report, you would get a dialog that allows you select and reuse any of the saved named parameter sets for this report:



Double-clicking any of the entries (or selecting an entry and clicking the OK button) would launch a dialog allowing you to reuse that set of saved parameter values or selectively change some of these saved parameter values.

This functionality was designed to address scenarios where reports with many parameters are used in one of several parameter patterns. By saving and naming these patterns, the user can reuse them.

Note: if you are a report developer, you can deliver these saved parameter patterns to a user machine by copying the [Named_Parameter_Sets] section in your DataLink_Viewer.ini file to the user's DataLink_Viewer.ini file.

Click to Change a Parameter Value

[Video Demo](#)

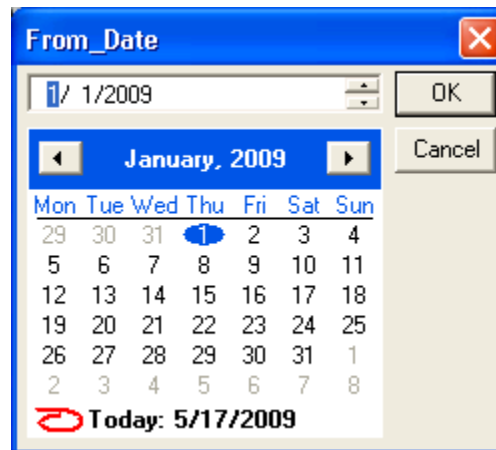
If you place a main report parameter on the report layout, when a DataLink Viewer user clicks on that parameter, DataLink Viewer prompts the user for just that parameter. That allows your users to simply and intuitively select just one parameter to refresh.

TOGGLING BETWEEN 2 PARAMETER VALUES

If a user clicks on a parameter that has **exactly 2 default values** (no custom values allowed), DataLink Viewer toggles the value of that parameter to the other value. This provides many useful behaviors. The sample report demonstrates toggling between viewing Profit or Revenue by simply clicking on the {?Metric} parameter. Another parameter allows toggling between expanding all top nodes or manually selecting which nodes to expand.

SPECIAL DATE PARAMETER DIALOG

If a user clicks on a single-value date parameter, DataLink Viewer presents an enhanced date entry/selection dialog. To quickly select a date, **double-click a date** on the calendar. Once the calendar is selected, **Page Up/Down** changes the month and **Ctrl+(Page Up/Down)** changes the year:



INPUT BOX DIALOG

If a user clicks on a single-value parameter with **less than 2 default values**, where custom values are allowed, DataLink Viewer presents a simple InputBox dialog, making it easier and faster for the user to change the parameter value.

LISTBOX DIALOG

If a user clicks on a single-value parameter **more than 2 default values**, where custom values are not allowed, DataLink Viewer presents a multi-row dialog, with one row for each default value designed for that parameter. This allows the user to quickly change the value for that parameter. To quickly select an item, the user can simply double-click it.

To quickly locate an item in the list, the user can type the first few characters and the list will automatically scroll to the first item starting with these characters. To quickly select an item, the user can simply double-click it.



CHANGING A PARAMETER BY CLICKING ON A RELATED FORMULA

In some cases, you may wish to invoke a parameter change by clicking on a related formula. For example, you may wish to embed the parameter inside some text, or perhaps the parameter may have an empty value, making it impossible to click on the parameter object by itself.

To use a formula as a proxy for clicking on a parameter, the formula name must start with "**DLV_Param_Click:**" followed by the parameter name you wish to change when a user clicks on that formula. For example, "DLV_Param_Click:Customers"

The formula itself may display any values you wish. For example, you may concatenate static text with the parameter value, or you may wish to return "No Value" if the parameter value is null, and a nicely formatted value if the parameter is not null..

Load ini Values into Parameters

Specially named parameters get loaded automatically from DataLink_Viewer.ini file entries. For example, you may use this with **parameters that change only once per quarter**. Another case could be a situation where you develop and sell reports (probably .rpt files converted to .rpz files) to clients in a vertical market. These reports are designed to work against a known database structure, but **each client may need to customize the reports with text elements, conditional formatting, or record selection criteria without changing the report design**. Or perhaps you wish to **restrict use of your reports to only paying customers by providing a license code** that must match (using secret logic) the company name in the customer's database.

Instructions:

First, add to the [Options] section in **DataLink_Viewer.ini** (in the application folder) an entry with a **key name** (Company) and value (Millet Software) of your choice. For example:

```
[Options]
...
Company=Millet Software
```

Then, add a **String Single-Value parameter** named “**DLV_INI_Option_KeyName**” to the report. DataLink Viewer automatically sets the value of such parameters to the value found for the Key Name under the [Options] section of DataLink_Viewer.ini. So, in the case above, the parameter value of **DLV_INI_Option_Company** would be set to “Millet Software”.

Notes:

- Within DLV, the **user never gets prompted** for the value of such parameters.
- If a matching key name can't be found, the parameter gets the value of: “Failed INI Lookup”
- You can have as many parameters like this as you wish, each with a different key name.
- To pass such parameters to subreports, create a formula in the main report that simply returns the value of the parameter. Then, pass that formula as a link to the subreport.
- The ini file used by this process is always the MASTER ini file (the one in the DataLink Viewer application folder), even if you redirect to a DataLink_Viewer.ini at a different location. This allows you to enforce this type of parameters from a centralized location.

Securing Reports against Unauthorized Use

If you wish to secure your reports against unauthorized use, you can provide authorized users a License Key string for the ini entry. Within the report (later distributed to clients as an .rpz file) you design a record selection criterion that returns true only if the license key matches the company name in the database. As a simple example, you could check the number of characters in the license key is equal to the length of the actual company name (in the database) plus the number of R's in that company's name.

If you don't have access to the company name in the database, or if you wish to issue per-machine licenses, use the technique described in the next section.

Load Machine/Company/Report Information into Parameters

DLV_MACHINE_NAME

If a report has a single-value string parameter called **DLV_Machine_Name**, DataLink Viewer automatically sets the value of such a parameter to the Windows Machine Name on which DataLink Viewer is running.

DLV_REGISTERED_COMPANY

If a report has a single-value string parameter called **DLV_Registered_Company**, DataLink Viewer automatically sets the value of such a parameter to the Company Name specified when the Windows operating system was installed..

In conjunction with the ability to load ini values into report parameters, this functionality allows report developers to distribute rpz files and restrict their use to only those where a license code matches the company to which Windows is registered.

DLV_HD_SERIAL_N

If a report has a single-value string parameter called **DLV_HD_Serial_N**, DataLink Viewer automatically sets the value of such a parameter to the serial number of the current hard drive.

In conjunction with the ability to load ini values into report parameters, this functionality allows report developers to distribute rpz files and restrict their use to only those where a license code matches the machine on which the report is running.

DLV_RPT_PATH

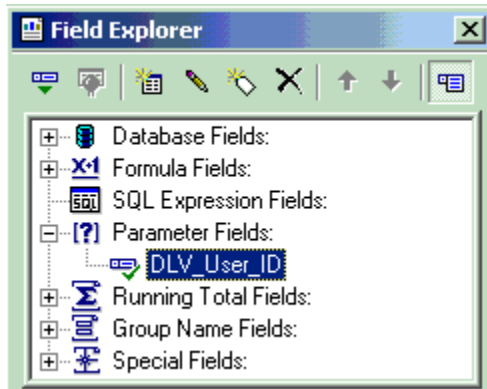
If a report has a single-value string parameter called **DLV_Rpt_Path**, DataLink Viewer automatically sets the value of such a parameter to the path to the rpt or rpz file. This is particularly useful in the case of rpz files.

Control Data Access according to PC Login

Assume you have several Sales Reps who need access only to their own sales information. DataLink Viewer can limit the data shown in a report according to who is logged in to the PC.

To achieve this functionality you need two elements:

1. The report must have a String parameter named “DLV_User_ID” as shown below.
DataLink Viewer automatically sets the value of such a parameter to the User ID who is currently logged in to the PC.



2. **The record selection formula must include a condition that restricts the data according to the User ID.** In most cases, this would be achieved by creating a new table in your database (or adding a new column to an existing table). For example, a new column in the Sales_Rep table (or a new table) can associate each Sales_Rep_ID with his/her User ID.

The record selection formula would include a condition such as:

{Sales_Rep.User_ID}=?DLV_User_ID

This would ensure that as each Sales Rep logs into their PC, DataLink Viewer will show them only the sales records associated with their own Sales_Rep_ID.

Note: using a table, you can use multiple records to map the same User_ID to multiple areas of responsibility (Customer_IDs, Product_IDs, Region_IDs).

This allows you to give a single user (for example, a regional manager who should be able to see information for all her Sales Reps) permissions to view information that is related to more than one entity.

In the table example on the right, **ixm7** is allowed to view sales information for all three products, while **ido** is allowed to view sales information only for Visual CUT.

Product Code	User_ID
Visual CUT	ixm7
CUT	ixm7
DataLink Viewer	ixm7
Visual CUT	ido
CUT	Joe
DataLink Viewer	Mike

Dynamic & Cascading Parameters

[Video Demo](#)

This section explains the special dynamic & cascading parameter functionality provided by DataLink Viewer. **When a parameter name matches a report name, DataLink Viewer uses that report as a parameter dialog.**

The first parameter in our sample report was named (by the report developer) as:

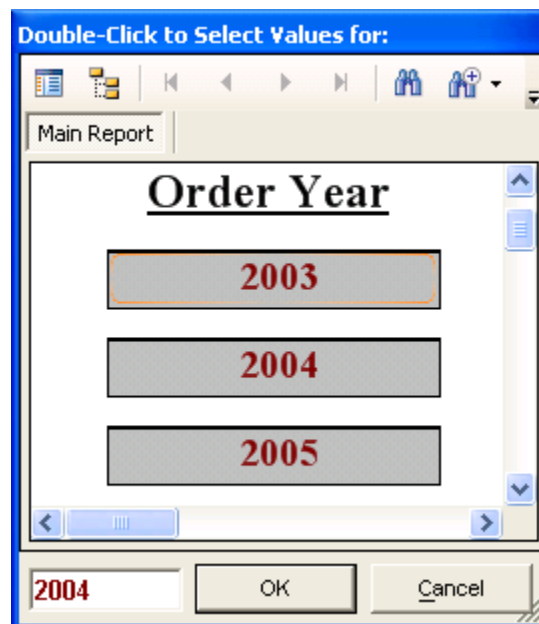
“Prompt_Order_Year.rpt”

DataLink Viewer detects that such a report exists and “links” to that report as the basis for prompting the user.

Note: if a prompting report doesn’t exist, the static parameter dialog provided by Crystal will be used. *DataLink Viewer* always tries to locate and use Live Prompts before using the static Crystal prompts.

RESPOND TO SINGLE-VALUE LIVE PROMPTS

DataLink Viewer detects that the **Prompt_Order_Year.rpt** parameter was designed to accept only a single value, and hence it presents you with the linked report inside a dialog that allows you to click and select only a single value:



Note that by linking to a live prompt report, the designer of the report doesn’t have to change the report design every year in order to add one more entry to the default value list of the static parameter. Instead, the prompt report dynamically presents all available years in the database.

LINK BACK IN A HIERARCHY OF LIVE PROMPTS

Once you select 2004 and hit OK, the viewer detects a second parameter in our sample report:

“**Prompt_Products.rpt**”

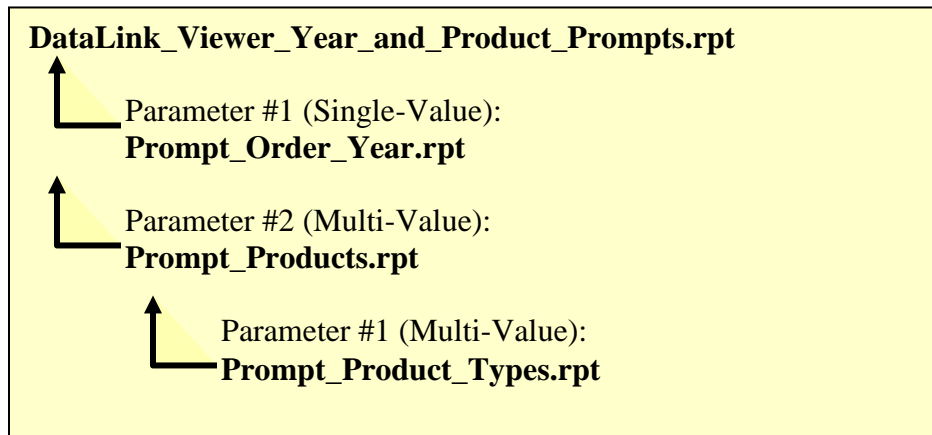
Again, *DataLink Viewer* detects that such a report exists and links to that report as the basis for prompting the User.

However, before presenting this report to the user, the viewer detects that the

Prompt_Products.rpt itself is designed with a parameter of its own:

“**Prompt_Product_Types.rpt**”

What we have here is a *hierarchy of links* whereby the values selected by the user in response to the **Prompt_Product_Types.rpt** parameter restrict the values available for selection in the **Prompt_Products.rpt** parameter:



***DataLink Viewer* always starts from the bottom of the hierarchy for each parameter!**

Hence, we are going to be prompted first for Product Types before we select Products within those Product Types.

RESPOND TO MULTI-VALUE LIVE PROMPTS

DataLink Viewer detects that the parameter called **Prompt_Product_Types.rpt** was designed to accept multiple values, and hence it presents you with the linked report



Prompt_Product_Types.rpt inside a dialog that allows you to click and select multiple values:

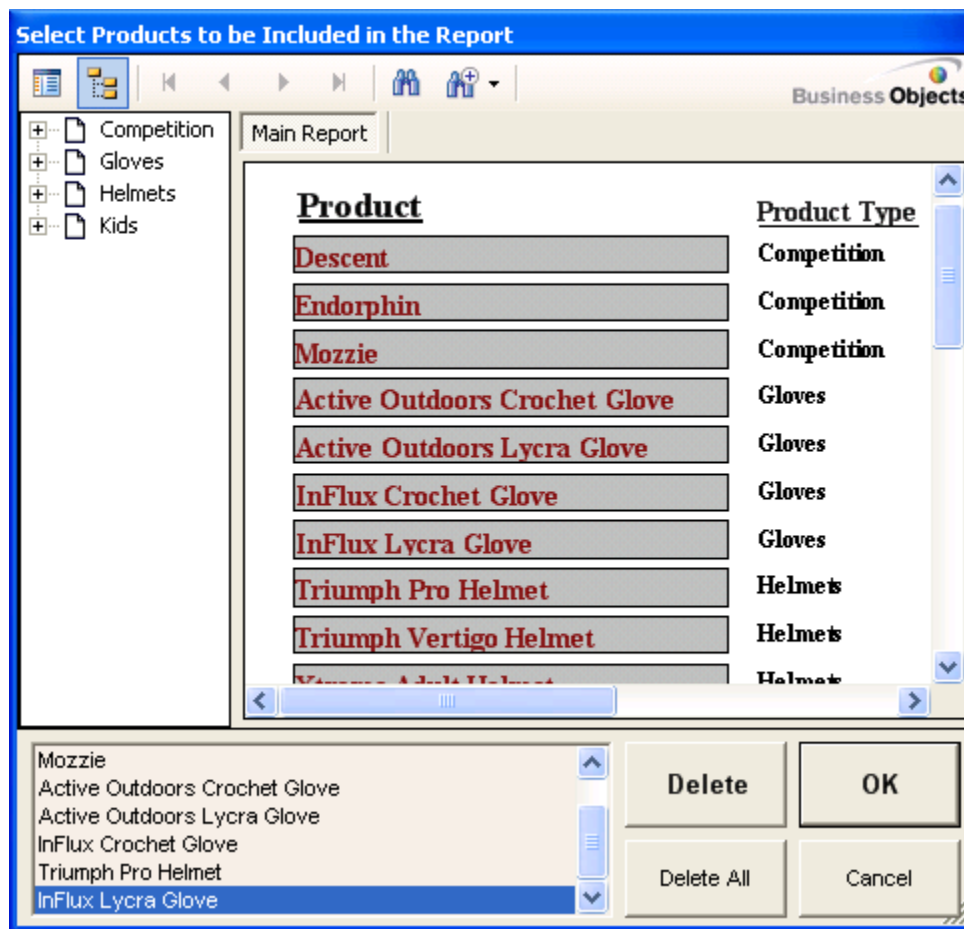
After clicking on *Competition* and *Helmets* to select those Product Types, click OK to progress to the Products selection within those Product Types.

MANAGE DISPLAY & DATA VALUES FOR LIVE PARAMETERS

In this particular situation, the **Product Type Name** is used for both **display value** (what gets shown to the user in the list of available values) as well as **data value** (what gets passed into Crystal as the parameter value). As demonstrated in the following step, *DataLink Viewer* allows you to present the user with meaningful product names as **display values** while passing Product IDs as the actual parameter **data value**.

Again, *DataLink Viewer* detects that the parameter called **Prompt_Products.rpt** was designed to accept multiple values, and hence it presents you with the linked report **Prompt_Products.rpt** inside a dialog that allows you to click and select multiple values. **Note that only Products within the Selected Product Types (in the previous step) are available for selection.**

This demonstrates the power of **cascading dynamic parameters**:



While you can click on any field/formula to add its text to the selection list, it's good practice to use color, font, and box effects to highlight the column you intend the user to click..

When you click on a field, the viewer always uses what you click on as the Display Value. However, if that field has a tooltip, the text of that tooltip is used as the parameter Data Value. After clicking on some product names, move your cursor over these items in the list of selected Display Values. Notice that a popup text reflects the underlying Data Value (Product IDs for

each product).

You can **resize** the dynamic parameter dialogs, select a **zoom** level, and turn the **group tree** on or off for each dynamic report. **DataLink Viewer remembers these settings** and will apply them the next time the same dynamic parameter dialog (for the same dynamic parameter report) is loaded.

VIEW THE REPORT

When you click OK, DataLink Viewer uses the selected values for both the “Prompt_Year.rpt” and “Prompt_Products.rpt” parameters to display the requested report:

The screenshot shows the DataLink Viewer interface with a report preview. The report title is "Product Sales by Type in 2004". Below the title, it displays "Competition [\$291,560]". A yellow highlighted box contains the instruction: "Double Click one of these Formulas to Launch *ProductType Catalog.Rpt* Passing to it **Competition** as a Parameter:". Below this are two formula boxes: "DLV_Run_Here:-v \"Product Type Catalog V12.rpt\" \"Parm1:Competition\"" and "DLV_Run:-v \"c:\Program Files\DataLinkViewer 12\Product Type Catalog V12.rpt\" \"Parm1:Competition\"".

Product	Revenue
Descent	134,057.18
Mozzie	102,477.22
Endorphin	55,025.87

Below the table is a bar chart titled "Revenue By Product" showing the revenue for each product. The bar for "Descent" is the tallest, followed by "Mozzie" and then "Endorphin".

The status bar at the bottom of the window shows "DataLink_Viewer_Year_and_Product_Prompts V12.rpt" and "3 Records".

Dynamic and Cascading Parameters (Developer Notes)

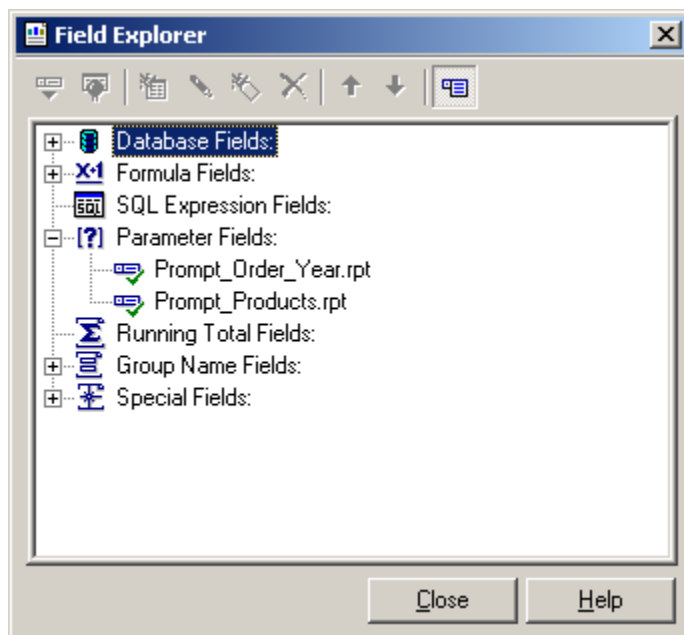
DataLink Viewer can be used to run existing Crystal reports without any design changes.

This section reviews how report designers can activate the extra functionality of linking to live prompt reports.

NAMING PARAMETERS

If you want to link a parameter to a live prompt report, you need to name the parameter as the report name. In our sample report, the live prompt reports and parameter names were:

“Prompt_Order_Year.rpt” and “Prompt_Products.rpt”:



DESIGNING LIVE PROMPT REPORTS

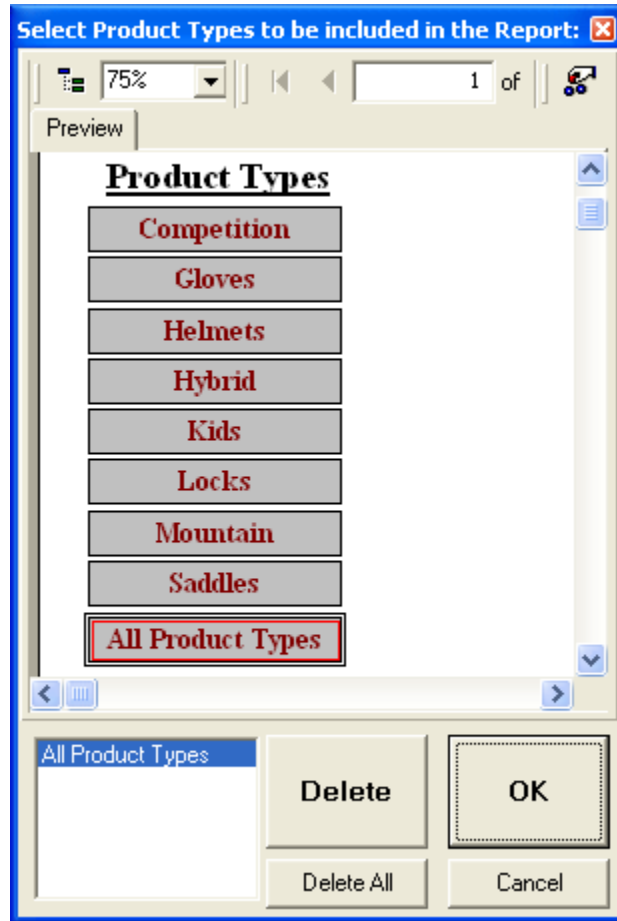
The reports acting as live prompts should be placed in the same directory as the report to be viewed or in the DataLink_Viewer directory.

These reports should contain at least one formula with the word “parameter” within its name. Formulas named in such a manner can be clicked and selected within the prompt viewer.

You can place a field or a formula on the report to let the user select a meaningful Display Value (e.g., “@Product_Name”). If you give that field or formula a tooltip text or expression, that value would be used as the parameter Data value.

IMPLEMENTING A “SELECT ALL” OPTION

If you run the report again and scroll to the bottom of the live Product Type selection prompt, you would notice that you can select the “All Product Types” option:



This was implemented by adding a formula to the report footer with a static text value of “All Product Types”. The desired effect is obtained by specifying the following record selection criterion in **Prompt_Products.rpt**:

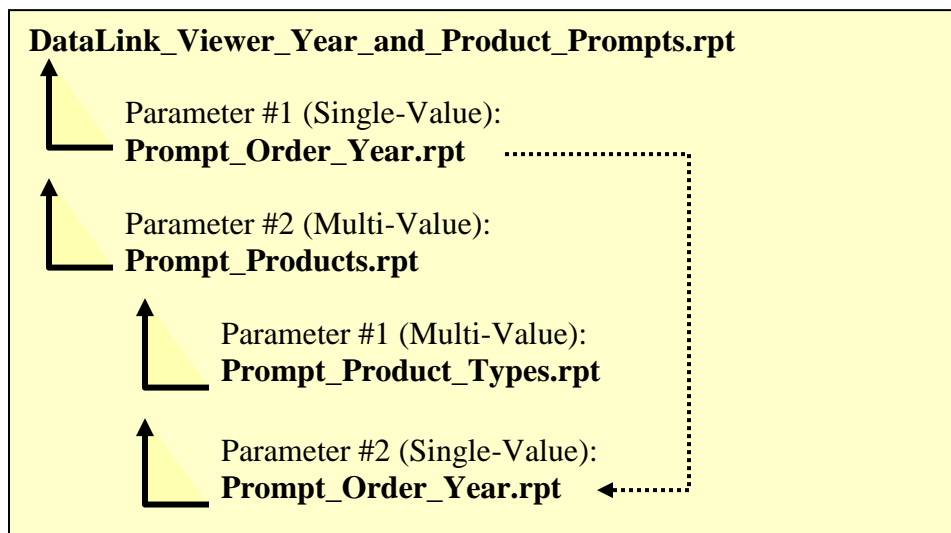
{Product_Type.Product Type Name} in {?Prompt_Product_Types.rpt}
OR
"All Product Types" in {?Prompt_Product_Types.rpt}

SHARING VALUES ACROSS CASCADING PARAMETERS

Parameter values are automatically reused if the same Parameter name is used more than once in the chain of cascading parameters.

The diagram below shows a situation where **Prompt_Order_Year.rpt** is used to first ask the user to specify which orders should be shown in the report. That same parameter is then also used to limit the dynamic list of products (**Prompt_Products.rpt**) such that only products that actually sold in that year are available for selection.

DataLink Viewer now reuses the year specified by the user the first time **Prompt_Order_Year.rpt** is invoked to avoid asking the user the same question twice.



Note: A **DataLink_Viewer.INI** file is automatically created the first time *DataLink Viewer* links to a live prompt report. To disable sharing parameter values from prior answers, open this ini file using any text editor and change the following entry from TRUE to FALSE:

```
[Options]  
DataLink_Parameters_Share_Prior_Answers=TRUE
```

REQUIRING A VALUE FOR A DYNAMIC PARAMETER

This can be achieved by following these guidelines:

1. Add a **string** formula with the name of: **{@DLV_PARAMETER_REQUIRED}** to the Report Footer (of the dynamic parameter report, not the main report).
2. The formula must **return a non-blank string** if you wish to enforce a required value for the parameter.
3. If the string returned by the formula is longer than 10 characters it would be used as the text of the message box in cases where the user clicks OK without specifying a value for the parameter. Otherwise, a standard message appears.
4. The formula, and even the report footer section as a whole can be suppressed.
5. You must place the "Page N of M" special field in the Page footer of the dynamic parameter report (even if that field or the page footer as a whole are suppressed).

FORCING USERS TO SELECT FROM THE LIST OF VALUES (NO DIRECT EDIT)

By default, dynamic parameter dialogs for single-value parameters allow the user to enter a value directly instead of clicking on the displayed list. **You can force the user to select from the list displayed by the dynamic parameter report** under two scenarios:

1. You include the text '**NoEdit**' in the parameter name.

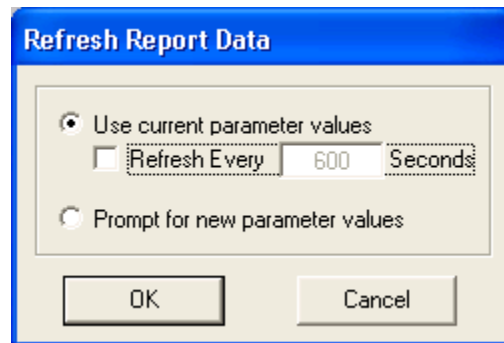
- or -

2. The user double-clicked an item in the dynamic parameter report and that item has an associated data value (tooltip expression). If the option to remember dynamic parameter values is turned on, from that point on that parameter will force the user to select from the list.

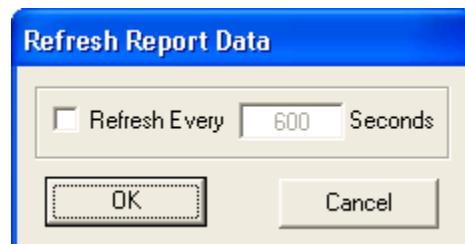
Auto-Refresh Reports

When clicking the Refresh button, you can specify that the report should automatically refresh every N seconds.

As shown in the following dialog, when a report has parameters, the Auto-Refresh option is enabled only when you elect to reuse the current parameter values:



When a report doesn't have parameters, the dialog is simpler and you can always elect to enable auto-refresh:



The auto-refresh process stops when you click the Refresh button again or when you select another report.

Auto_Refresh Command Line Argument

You can also trigger auto-refresh by using an "Auto_Refresh" argument when invoking a report from a command line. Here's how you would trigger a viewing of a report and auto-refreshing it every 5 minutes (300 seconds):

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "Auto_Refresh:300"
```

Auto_Page_and_Refresh Command Line Argument

Using this command line argument, you can request the viewer to automatically advance to the next page every N seconds. After getting to the last page, the viewer will next move to the first page and also refresh the report data. The cycle then continues. Here's how you would trigger the process with 20 seconds as the time increment:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "Auto_Page_and_Refresh:20"
```

Note: to pause the auto-page & refresh process, double-click on the report -- Double-click again to resume.

ViewMode Command Line Argument (Remove Toolbar/Status bar)

When you trigger a report from a command line using a -v (View Only window), you can also elect to remove the toolbar and/or the status bar in order to maximize the space available for the report itself.

The command line argument for controlling this behavior is "ViewMode" and it has 3 possible values:

- ... "ViewMode:**NoStatusBar**" - only the status bar at the bottom of the window is removed
- ... "ViewMode:**NoToolBar**" - only the toolbar at the top of the window is removed
- ... "ViewMode:**NoBars**" - both the toolbar and the status bar are removed

A typical scenario for using this option is in combination with the Auto_Refresh command line argument. For example:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "Auto_Refresh:300" "ViewMode:NoBars"
```

Cycling Through Several Auto-Refreshed Reports

Assume you want your monitor to cycle through 2 different auto-refreshed reports every 15 seconds. You can take advantage of the fact that as a report gets auto-refreshed its window becomes the active one. So, you can trigger auto-refresh every 30 seconds for both reports, but separate the start times by 15 seconds.

Here's how you can automate the whole process:

Step 1: Use notepad to create a batch file that can be called from within another batch file to delay processing by a given number of second. Call it **WAIT.bat** (if you'd like to understand the logic, see: <http://malektips.com/dos0017.html>):

```
@ping 127.0.0.1 -n 2 -w 1000 > nul  
@ping 127.0.0.1 -n %1% -w 1000> nul
```

Step 2: Use notepad to create the following text file. Call it **invis.vbs** (this allows us to call 1 report and progress to the next line in a batch file (no blocking) as well as avoid the ugly DOS window):

```
CreateObject("Wscript.Shell").Run """" & WScript.Arguments(0) & """" , 0, False
```

Step 3: Use notepad to create one batch file for each report to trigger its viewing and auto-refreshing. Here are two batch files, one for a report with a parameter and one for a report without one:

Report_1.bat

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v "C:\DLV\Report_1.rpt"  
"Parm1:2004" "Auto_Refresh:30"
```

Report_2.bat

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v "C:\DLV\Report_2.rpt"  
"Auto_Refresh:30"
```

Step 4: Use notepad to create a batch file, which silently calls the two reports, inserting a 15 seconds wait between the calls. Double-click or schedule that batch file to start the processing:

```
wscript.exe "C:\DLV\invis.vbs" "C:\DLV\Report_1.bat"  
CALL WAIT 15  
wscript.exe "C:\DLV\invis.vbs" "C:\DLV\Report_2.bat"
```

Click to Set Formula Value

[Video Demo](#)

DataLink Viewer reacts to user clicks on specially named formulas by setting the value of another formula. For example, if a user clicks on a formula called `{@DLVSet_1:Sort1}` that has as its string value the text "Region" it will attempt to locate the `{@Sort1}` formula and set its string value to "Region". If, the user clicks on another formula called `{@DLVSet_2:Sort1}` that has as its string value the text "Country" it will attempt to locate the `{@Sort1}` formula and set its string value to "Country".

The name of the clicked formula has 4 parts:

1. **DLVSet** this is a constant that identifies the formula to DataLink Viewer as a click source.
2. Any number (not limited to a single digit) (**1** or **2** in the examples above). This allows for multiple formulas, each with a different string value, to set the string value of the same target formula
3. a colon (:)
4. The name of the target formula (Sort1 in the examples above) whose string value would be set to the string value of the clicked formula.

With some creativity, this allows you to design reports that react to user clicks in useful ways.

Click Column Headers to Re-Sort The Report

As a demonstration of what can be achieved with the "Click to Set a Formula Value" functionality, the sample report (DLV_Column_Header_Sort) reacts to clicks on column headers by resorting the report based on the clicked column header. A short video demonstration is available at: http://www.milletsoftware.com/Download/DLV_Sort_By_Click.wmv

The four column headers are clickable formulas:

1. `{@DLVSet_1:Sort1}` (with a string value of "Customer Name")
2. `{@DLVSet_2:Sort1}` (with a string value of "Region")
3. `{@DLVSet_3:Sort1}` (with a string value of "Country")
4. `{@DLVSet_4:Sort1}` (with a string value of "Postal Code")

The target formula `{@Sort1}` starts with a design-time value of "Customer Name" and as its value gets changed due to user clicks, the `{@SortBy1}` formula reacts by returning one of the 4 database columns (Customer Name, Region, Country or Postal Code) . This in turn changes how the report groups are sorted.

Note: **group** rather than **record** sorting is used in this sample report because the Crystal runtime components don't re-sort records unless the report data is refreshed from the database. The Group re-sort occurs all on the client-side without hitting the database again.

In-Place Drill-Down

[Video Demo](#)

Crystal's Drill-Down functionality is limited to opening the detail in a new tab. However, in many cases users prefer to expand/collapse detail within the main report tab. By adding to the report a few formulas and using them in the suppress expression of detail sections, DataLink Viewer allows you to achieve In-Place Drill-Down functionality.


A report may initially preview with all details collapsed, like this:












Product Type	Revenue	Days To Ship
+ Competition	\$1,970,125	2.8
+ Locks	\$5,954	2.8
+ Helmets	\$49,985	2.8
+ Mountain	\$495,540	2.9
+ Gloves	\$12,166	2.9
+ Kids	\$51,696	2.9
+ Saddles	\$10,206	2.9
+ Hybrid	\$265,533	3.2

3/27/2006 3:14 PM DLV_In_Place_Drill_Down_11.rpt 2,616 Records

The "buttons" are actually a formula that return a plus or a minus sign. When the user clicks on the these "buttons" the level-2 detail is revealed:

Product Type	Revenue	Days To Ship																
+ Competition	\$1,970,125	2.8																
+ Locks	\$5,954	2.8																
+ Helmets	\$49,985	2.8																
- Mountain	\$495,540	2.9																
<table border="1"> <thead> <tr> <th>Product</th> <th>Supplier</th> <th>Revenue</th> <th>Days To Ship</th> </tr> </thead> <tbody> <tr> <td>+ SlickRock</td> <td>Craze</td> <td>\$185,897</td> <td>2.8</td> </tr> <tr> <td>+ Nicros</td> <td>Craze</td> <td>\$147,790</td> <td>2.8</td> </tr> <tr> <td>+ Rapel</td> <td>Craze</td> <td>\$161,854</td> <td>2.9</td> </tr> </tbody> </table>			Product	Supplier	Revenue	Days To Ship	+ SlickRock	Craze	\$185,897	2.8	+ Nicros	Craze	\$147,790	2.8	+ Rapel	Craze	\$161,854	2.9
Product	Supplier	Revenue	Days To Ship															
+ SlickRock	Craze	\$185,897	2.8															
+ Nicros	Craze	\$147,790	2.8															
+ Rapel	Craze	\$161,854	2.9															
+ Gloves	\$12,166	2.9																
+ Kids	\$51,696	2.9																

And when a  "button" (another formula) in level-2 is clicked, level-3 detail is revealed:

Product Type		Revenue	Days To Ship	
	Competition	\$1,970,125	2.8	
	Locks	\$5,954	2.8	
	Helmets	\$49,985	2.8	
	Mountain	\$495,540	2.9	
Product		Supplier	Revenue	Days To Ship
	SlickRock	Craze	\$185,897	2.8
	Nicros	Craze	\$147,790	2.8
Customer		Country	Revenue	Days To Ship
	Sports Fever	New Zealand	\$330	11.0
	Blazing Bikes	USA	\$1,319	8.0
	Piccolo	Austria	\$990	8.0
	Bikes R Us	USA	\$313	6.0
	Bike-A-Holics Anonymous	USA	\$957	5.5
	Others	USA	\$143,881	2.7
	Rapel	Craze	\$161,854	2.9
	Gloves	\$12,166	2.9	
	Kids	\$51,696	2.9	
	Saddles	\$10,206	2.9	
	Hybrid	\$265,533	3.2	

You are not limited to 1 drill-down path -- several branches can be expanded at the same time.

see video demo at: http://www.milletsoftware.com/Download/DLV_InPlace_DrillDown.wmv

The installation of DataLink Viewer includes a sample report (for example, DLV_In_Place_Drill_Down_12.rpt if you installed version 12). To learn how to create your own report with this functionality, open the sample report and look at the comments and expressions in:



5. **@DLV_Expand_Button_1** and **@DLV_Expand_Button_2** formulas
6. **@DLV_Current_Group_1** and **@DLV_Current_Group_2** formulas
7. **@DLV_Expanded_Group_List** formula
8. Suppress expressions for the sections:
 - a. **GH1b** (acts as column headers for level 2) and **GH2a**
 - b. **GH2b** (acts as column headers for level 3) and **GH3**

The same approach can be extended to any number of group levels.

This functionality is enabled only in the main report view (not in Crystal's drill-down tabs).

Note: using my CUT Light UFL, you can "persist" to an ini file the information in **@DLV_Expanded_Group_List** so that the report "remembers" its collapse/expanded state.

Converting Section Double-Click to In-Place Drill-Down

When you implement In-Place Drill-Down, as discussed above, you may wish to allow the user to trigger this behavior by double-clicking anywhere in the section (rather than just by clicking the  /  @DLV_Expand_Button_N formula placed in that section.

To override the default double-click behavior in Crystal (drill-down to a new tab) and divert the double-click action to an In-Place Drill-Down, simply use all Upper Case when naming the button formula: for example instead of

@DLV_Expand_Button_1



use

@DLV_EXPAND_BUTTON_1 .

Note: Verify that if you move the mouse cursor over the formula placed in the report section, the correct upper/lower case is shown. If you already created the formula and you wish to change it to upper case, you may need to propagate that change to the actual formula object in the report section by copying the formula (Ctrl-drag) and then deleting the old clone.


Note 2: if only the 2nd click on the button is recognized, you can correct that problem using the same procedure as above: copy the button formula (using Ctrl-drag) and then delete the old clone.

Hiding the / Buttons

If you use an upper case formula name for any of your @DLV_Expand_Button_N formulas, you can hide the formula by using the section background color as the font color. This would allow you to use double-clicks exclusively for In-Place Drill-Down actions (the user will not see the  /  Buttons).

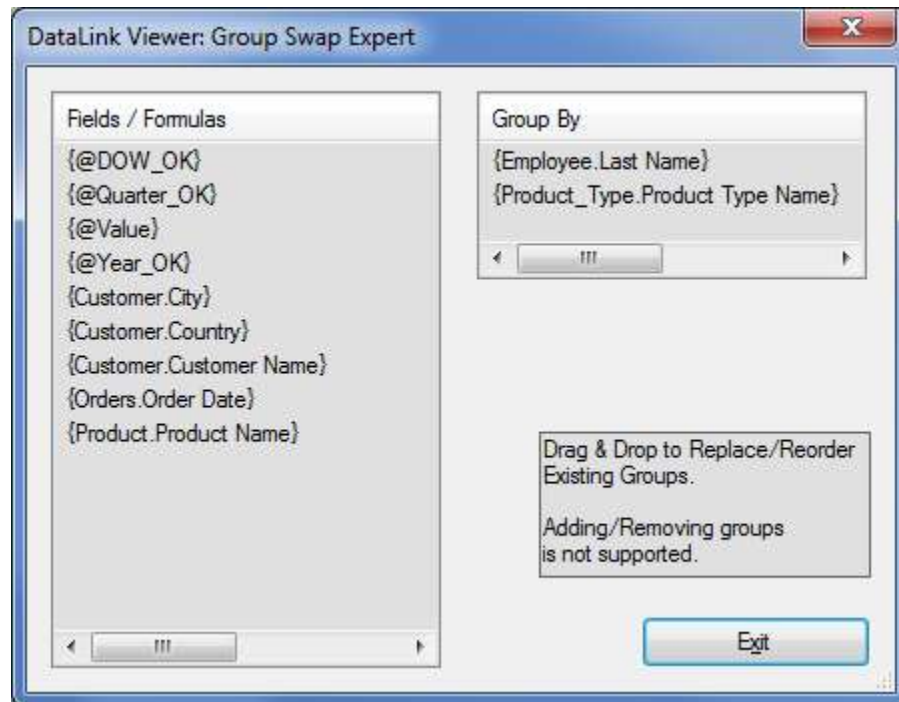
Dynamic Grouping (Group Swap Expert)

[Video Demo](#)

When previewing a Grouped report, if you click  or press **Ctrl-G**, the Group Swap Expert dialog open up. Using Drag & Drop, **you can change and reorder the fields/formulas used to group the report**. For example, instead of Grouping the report by Employee and by Product Type, you may group the report by Country and Year.

The grouping change **doesn't require hitting the database again, so it is very quick**.

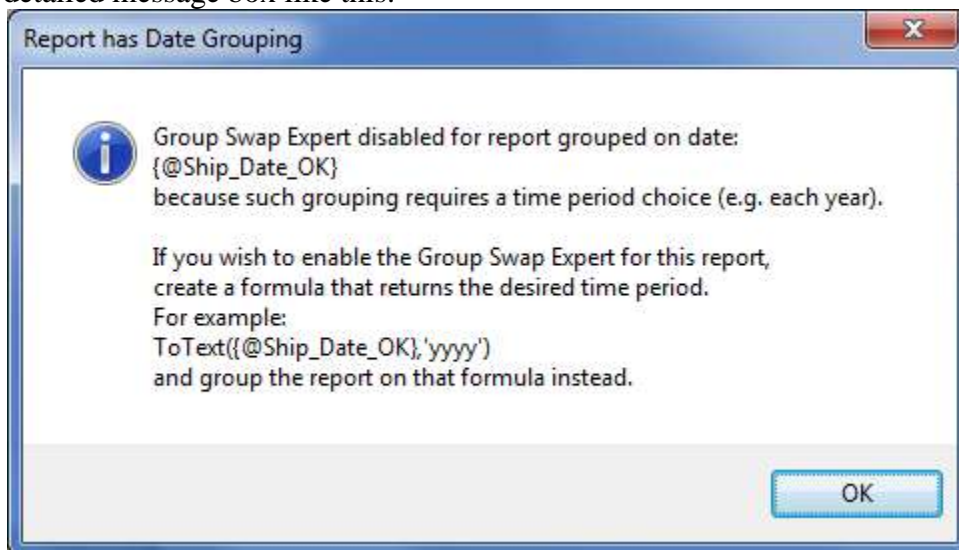
DataLink Viewer also adjusts all formulas using group summaries, as well as the Group Selection formula, to reflect the new grouping.



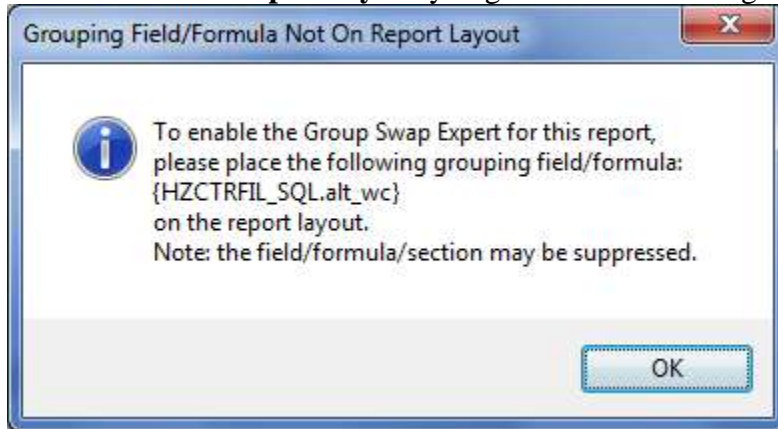
This feature avoids the need to create parameters and formulas in cases where you wish to provide dynamic grouping functionality. A video demonstration is available.

Notes:

1. If you try to invoke the Group Swap expert for a **report grouped on a date** you get a detailed message box like this:



2. For the same reason, **date fields/formulas are not listed as candidates for change of grouping.**
3. If you try to invoke the Group Swap expert for a **report grouped on a field/formula that is not on the report layout** you get a detailed message box like this:




4. To make a field/formula participate in this dialog, it must be placed on the report layout. Its name must not end with “**NOK**” (an indication it’s Not OK to include it) If the field/formula is suppressed or is placed in a suppressed section, it gets loaded into the list of candidate Fields/Formulas only if it’s name ends with “**OK**”. This approach is designed to avoid exposing suppressed information.
5. Conditional formatting formulas using group summaries are not adjusted.
6. Groups sorted Descending by total field may become sorted Ascending. To avoid that behavior, use a TopN sort (with a large N) instead.
7. You may disable this functionality by adding this to the master DataLink_Viewer.ini


```
[Options]
Disable_Group_Swap=True
```

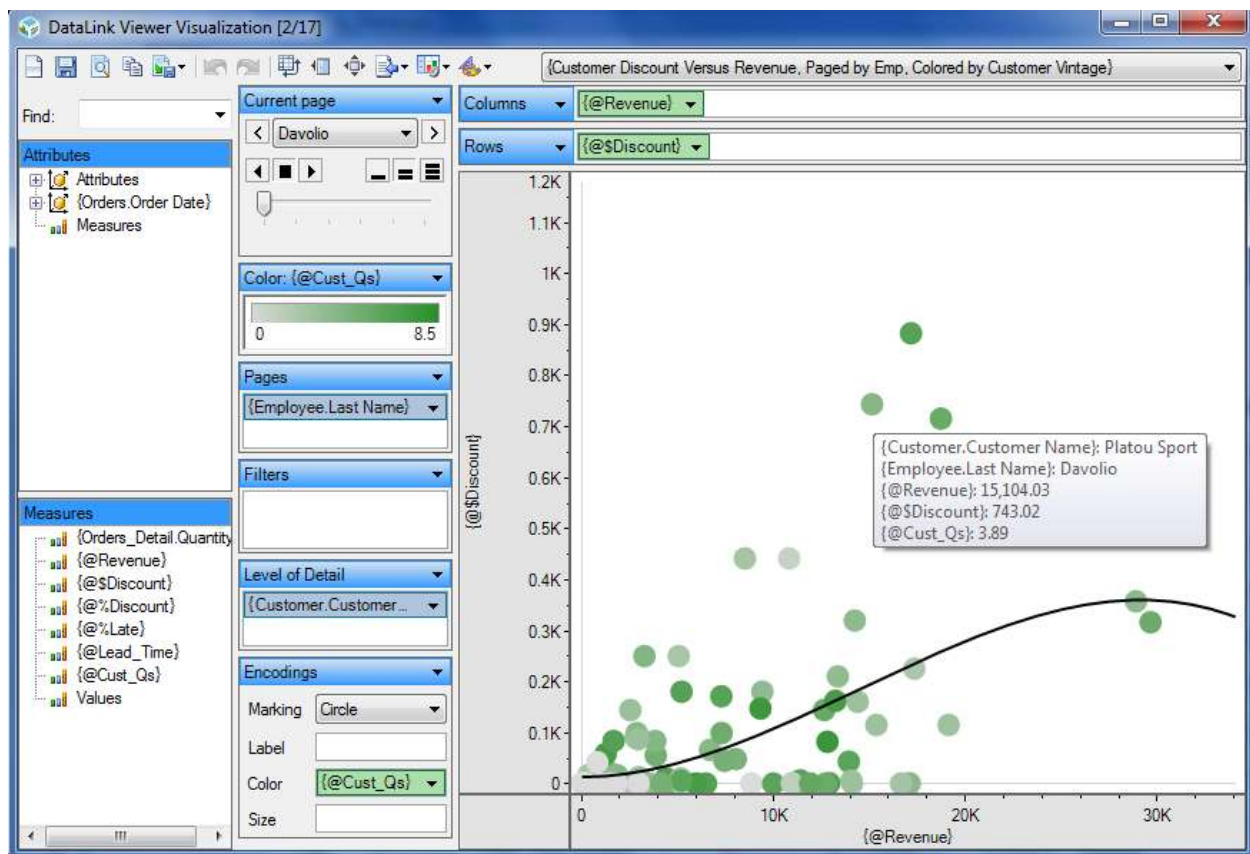
Optional Technique for Gaining Formula Access to the Dynamic Grouping:

If the report contains formulas names as: **DLV_Group_L1**, **DLV_Group_L2**, etc. DataLink Viewer will set the expression of each of these formulas to return the grouping Field/Formula at that level after applying group swaps. For example, you can design the report to have **{Employee.Last Name}** as the formula expression for **DLV_Group_L1** (reflecting the initial design of the report). If the user swaps **{Customer.Country}** for the level 1 grouping, DLV changes the expression in the **DLV_Group_L1** from **{Employee.Last Name}** to **{Customer.Country}**. **This allows you to build Advanced Charts (not limited to the Grouping Hierarchy) and CrossTabs that reflect the dynamic grouping.** The sample report (DLV_2011_Change_Group_Demo.rpt) demonstrates this feature.

Data Visualizer

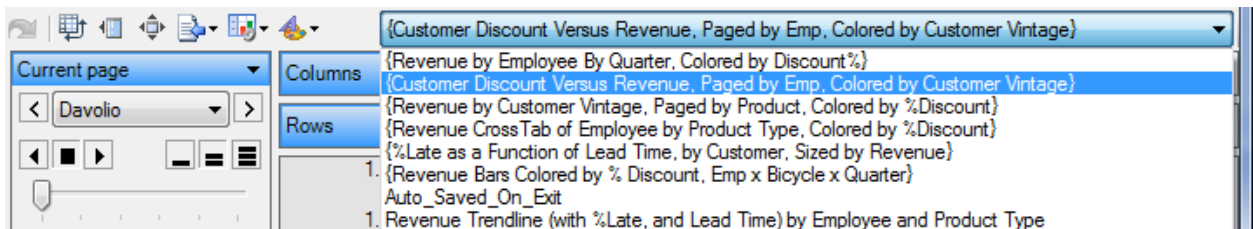
When previewing a report, clicking the  toolbar button loads the report data into a powerful data visualization tool. Similar to a tool called Tableau (which costs ~\$1,000/desktop) the idea is to go beyond regular pivot tables or charts. The user can drag-and-drop to display and drill-down into data patterns using effects such as multiple data panels, animation, filtering, sorting, coloring, sizing, shapes, trend lines, and filters.

DataLink Viewer takes care of loading the report data into the visualizer, and mapping it into Measures (numeric columns), Attributes, and Dates. Dates are treated as special attributes with Year, Quarter, Month, and day hierarchy. Users can then create, save, and reuse visualization layouts. Several short [video tutorials](#) are available on how users can build visualization layouts.



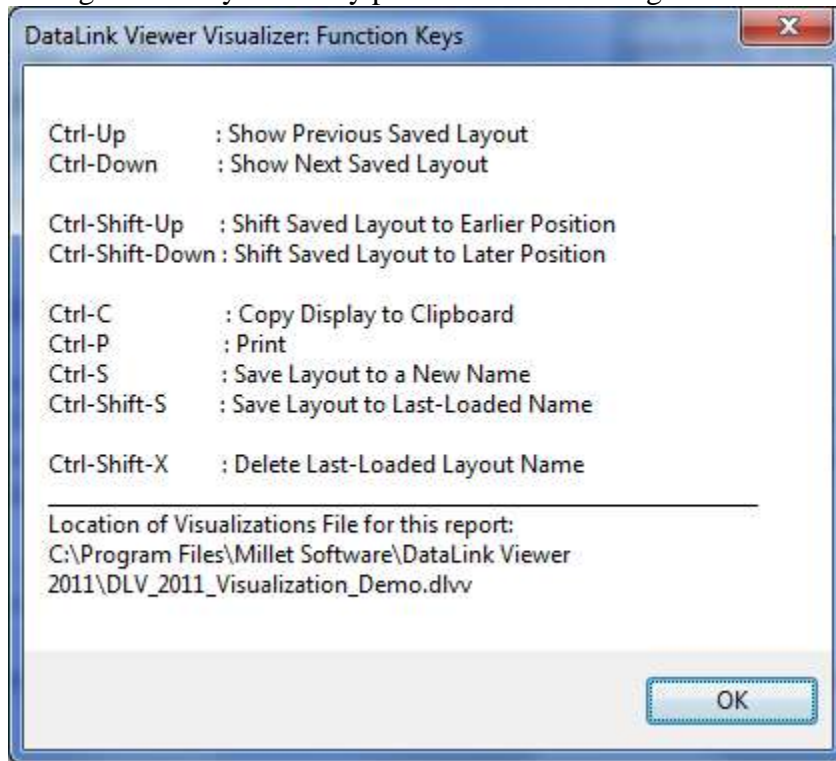
Notes:

- Since the visualizer takes care of generating summaries from the data, it is best to use it with detail (rather than summary) reports.
- To get loaded into the visualizer, a field/formula must be placed on the report layout. Its name must not end with “**_NOK**” (an indication it’s Not OK to include it). If the field/formula is suppressed or is placed in a suppressed section, it gets loaded into the list of candidate Fields/Formulas only if it’s name ends with “**_OK**”. This is designed to avoid exposing suppressed information.
- You may disable this functionality by adding this to the master DataLink_Viewer.ini
[Options]
Disable_Visualization=True
- When a user launches a report that has save visualization layouts, you may want DataLink Viewer to immediately launch the visualization, instead of the report preview. You can achieve that behavior by going into the DataLink Viewer Options dialog, Launch tab, and selecting a Visualization Action:
 - a) None
 - b) Keep Preview and Launch Visualization, or
 - c) Minimize Preview and Launch Visualization
- After creating a Visualization layout you can name and save it. What gets saved is only the layout, not the data. So each time you load report data, the layout applies to the new data.
- All Visualizations get saved into a single .dlvv file, located where the report file is located. This allows you to easily distribute the visualizations with your reports.
- When packaging an **rpz file**, a new checkbox option allows you to embed the .dlvv file inside the rpz file. This makes it easy for professional report writers to develop, protect, and deliver not only reports but also data visualizations to their clients.
- When loading the visualizer for a report that has saved visualization layouts, the 1st layout gets applied to the current data automatically. A drop-down provides access to all the other named visualizations:



In the example above, the top 6 layouts are surrounded with curly brackets {...}, indicating these visualizations were embedded inside the rpz file. The next 2 items were added by the user and are managed in a local .dlvv file. Only non-embedded entries can be added, deleted, renamed, and sorted by the user.

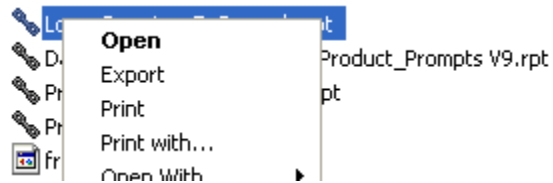
- Hitting the F1 keyboard key provides the following reminders about useful function keys:



Launching Reports & Command Line API

Launch Reports from File Explorer

DataLink Viewer allows you to launch a report from the Windows File Explorer by Right-Clicking a **.rpt** or **.rpz** file. The typical File Explorer menu provides the following extra options:



Open: launches a report to a Preview window.

Export: exports the report to a user selected export format and disk file.

Print: prints the report to the Default printer.

Print with...: prints the report to a user-selected printer.

These options allow users to invoke exporting or printing of a report without previewing it. This means users don't have to wait for the preview to complete before providing exporting or printing choices. Similar choices are available from the full user interface in *DataLink Viewer* (by right-clicking a report row in the 1st-tab grid). However, some users may prefer to use File Explorer as a launch mechanism without starting *DataLink Viewer*.

Note: *DataLink Viewer* associates these file explorer actions with **.rpt** and **.rpz** files only on PCs where these extensions are not already associated with other applications. This means that on a machine where Crystal Reports is already installed (where **.rpt** files are already associated with the Crystal application) you will not see these right-click menu options in File Explorer.

Launch Reports from Command Lines

DataLink Viewer allows you to launch a report from another application or from a shortcut using a command line.

The general structure of the command line is:

- 1) the full path and name of the **DataLink Viewer executable**
- 2) **-S** or **-V** as a processing flag indicating we are requesting a report to be shown/viewed
 - S** shows the report and *allows selection of other reports* via the Report Selection Tab
 - V** is a *View Only* option that *allows only viewing of the specified report* (the Report Selection Tab is not available to the user)
- 3) the **report path and file name**
- 4) Optional **Parameter value arguments** supplied as "**ParmN:Value**" (where N is the position of the parameter in the parameter list within the Crystal report)
- 5) Optional **User ID** and **Password arguments**
- 6) Other optional arguments: Export, Printer, Print_Copies, ODBC_DSN, ODBC_DSN_From_To, Oracle_Server, Connect_To_SQLOLEDB, Auto_Refresh, ViewMode, Disable_Print_Button, Disable_Print_ThisPage_Button, Disable_Print_Quick_Button, Disable_Export_Button, Disable_Search_Button, Disable_Refresh_Button, DB_Path_Use_Default, Disable_Group_Swap

Examples:

1. Invoke viewing of a sample report while **passing 1997 as the 1st parameter value:**

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe"  
-s "C:\Temp\DataLink_Viewer_Year_and_Product_Prompts.rpt" "Parm1:1997"
```

2. Invoke viewing of a secure report while **passing User_ID and Password:**

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe"  
-s "C:\temp\Some Secure Report.rpt" "user_id:dba" "password:sql"
```

3. Invoke viewing of a report with only Preview tab (no Select Report tab) and specify an ODBC data source. This is useful in cases where you have multiple database with the same structure. You create several ODBC DSNs and can control via a command line which database is used:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe"  
-v "C:\temp\MyReport.rpt" "ODBC_DSN:Company1"
```

Call DataLink Viewer from Another Application

4. Here's a code example of invoking DLV from a visual basic application with only a Preview tab (no Select Report tab) and specifying a parameter value. Note that double quotes are "escaped" by using

""

instead of

"

```
DimRetVal
ls_temp = "C:\Program Files\Millet Software\DataLink Viewer
2011\DataLink_Viewer_2011.exe " & _
"-v ""c:\temp\Lab2_Chase.rpt"" ""Parm1:800"""
```

RetVal = Shell(ls_temp)

5. Here's a code example for invoking DLV from a vba event and dynamically setting the report path, name, and parameter value:

```
Private Sub Combo0_AfterUpdate()
Dim rs As Object
Set rs = Me.Recordset.Clone
rs.FindFirst "[txtReportName] = '" & Me![Combo0] & "'"
If Not rs.EOF Then Me.Bookmark = rs.Bookmark

Dim myreport As String
Dim stAppName As String
Dim myvalret As String
' me.fullrep is a field that concatenates the report path and name
myreport = Me.fullrep
myvalret = Str(MyCaseno)
stAppName = "C:\Program Files\Millet Software\DataLink Viewer
2011\DataLink_Viewer_2011.exe" & _
"-s """" & myreport & """" ""Parm1:" & myvalret & """""
```

DoCmd.Close
Call Shell(stAppName, 1)
End Sub

Command Line Arguments for Parameter Values

Since parameter values can have different data types and structures (discrete, multi-value, range, mixed), this section explains how you can specify parameter values via command lines. All arguments must be **enclosed in double quotes** and **separated by a single space**.

A numeric or string parameter would be specified as:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\MyReport.rpt" "Parm1:1998"
```

A date parameter would be specified as:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\MyReport.rpt" "Parm1:3/10/2003"
```

DataLink Viewer handles the data type conversion to match the parameter data type. The syntax is constructed as the word “Parm”, followed by the number of the parameter (according to the order of parameters shown in Crystal), followed by a colon and the value.

Here’s how you would specify the 1st and 3rd parameter values:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\Program Files\DataLink Viewer 12\MyReport.rpt" "Parm1:1998" "Parm3:Ido  
Millet"
```

RANGE AND MULTI-VALUE PARAMETERS

DataLink Viewer supports all parameter types including multi-value, range, and mixed parameters. A multi-value discrete parameter value is specified as follows:

```
... "Parm1:Competition:::Gloves:::Helmets"
```

A range parameter (in this case a date range) is specified as follows:

```
... "Parm1:7/15/1996>>>7/15/2003>>>3"
```

The **3** at the end indicates the start and end points are included.

A **0** at the end would indicate the start and end points are NOT included.

A **2** at the end would indicate the start point is included and the end point is not.

A **1** at the end would indicate the start point is not included and the end point is.

Add 4 to these values if there is no Upper Bound.

Add 8 to these values if there is no Lower Bound.

OPTIONAL PARAMETERS WITH NO VALUE

To specify no value for an optional parameter, provide a blank value (e.g., “Parm2:”). **HasValue()** function, within the Crystal report, would then return False, as if the user didn’t specify any value for the optional parameter.

NULL VALUES

Null parameter values (for stored procedures) are specified in command lines by using the constant [VC_NULL]. For example, to specify that the first parameter value is null, use: "Parm1:[VC_NULL]"

IGNORING SAVED PARAMETER VALUES

In order to ignore saved parameter values, use the following command line argument:

"Ignore_Saved_Parameter_Values:[ALL]"

The user would then get prompted for unspecified parameter values, skipping the saved parameter values dialog.

Command Line Arguments for Triggering Exporting

Using the **"Export"** command line argument you can trigger exporting for a given report so the user doesn't need to preview the report or see the DataLink Viewer user interface (although the user will see an export options dialog). The syntax is constructed as the word **"Export"**, followed by a colon and the word **"Dialog"**

Here's how you would trigger an export dialog for a given report:

```
"C:\Program Files\DataLink Viewer 12\DataLink_Viewer_12.exe" -v "C:\temp\Report.rpt"  
"user_id:dba" "password:sql" "Parm1:Gloves" "Export:Dialog"
```

Command Line Arguments for Triggering Printing

Using the "**Printer**" command line argument you can trigger printing for a given report in an unattended mode (the user doesn't need to preview the report or see the DataLink Viewer user interface).

The syntax is constructed as the word "**Printer**", followed by a colon and:

- the word "**Default**" if the report should be sent to the default printer, or
- the word "**Dialog**" if the user should be prompted to select a printer, or
- the **printer name**, if the report should be sent to a specific printer

Here's how you would send a report to the **default printer**:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "user_id:dba" "password:sql" "Parm1:Gloves"  
"Printer:Default"
```

Here's how you would send a report to a **printer selected by the user**:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "user_id:dba" "password:sql" "Parm1:Gloves"  
"Printer:Dialog"
```

Here's how you would send a report to a **specified printer**:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "user_id:dba" "password:sql" "Parm1:Gloves"  
"Printer:\\Srv1\Laser1"
```

SPECIFYING NUMBER OF COPIES

To control the number of printed copies, you can use a "**Print_Copies**" argument. The syntax is constructed as the word "**Print_Copies**", followed by a colon and the number of desired copies. For example:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\Report.rpt" "Printer:\\Srv1\Laser1" "Print_Copies:3"
```

SCHEDULING PRINTING

Since you can trigger printing using the Printer command line argument, you can use the Windows Task Scheduler (or any other scheduler/application) to trigger printing of the report. For detailed instructions on how to use the free windows task scheduler, see the section on "Scheduling" in my **Visual CUT User Manual** (Visual CUT provides much more powerful report scheduling options) at: www.milletsoftware.com/visualcutManual.htm

SCHEDULING PRINTING FOR MULTIPLE REPORTS

If you need to schedule printing of multiple reports at the same time, the best approach is to place all command lines into a single batch file. Then, schedule the batch file in the free Windows Task Scheduler as described above.

Argument for Documenting File Locations and Report Parameters

The **Document2INI** command line argument allows other applications to request information from DataLink Viewer about:

- key file locations (DataLink_Viewer.ini, ReportList.txt)
- design of parameters for a given report

The command line call looks like this (all in 1 line):

“C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe”

“**Docunebt2ini**:c:\My_Reports\my_Report.rpt>>c:\temp\MyFile.ini”

The two elements after the **Docunebt2ini** argument name and the colon are:

1. the rpt file containing the parameters to document. If the rpt file doesn't exist, only the key file locations get updated in the ini file.
2. The ini file that should be created/updated with the documentation

If the specified rpt file doesn't exist, only file locations are set in the ini file. For example:

[File_Locations]

Master_Ini_File=C:\ProgramData\MilletSoftware\DLV_2011\DataLink_Viewer.ini

Slave_Ini_File= C:\ProgramData\MilletSoftware\DLV_2011\DataLink_Viewer.ini

Report_List= C:\ProgramData\MilletSoftware\DLV_2011\ReportList.txt

If the specified report does exist, information about report parameter gets added to the ini file, providing summary info as well as dedicated section for each parameter based on its position in the report:

[Parameters]

c:\temp\test.rpt=1||{?City}||String||False||False----2||{?Country}||String||False||False

[1]

Name={?City}

InUse=True

Type=String

PromptText=Enter City:

SingleValue=False

RangeValue=False

AllowEditing=False

Optional=False

[2]

Name={?Country}

InUse=True

Type=String

PromptText=Enter Country:

SingleValue=False

RangeValue=False

AllowEditing=False

Optional=False

Launch a Report while Viewing another Report

DataLink Viewer allows you to view a report and launch another report into its own Viewer window or in the current Preview tab by double-clicking a report section containing a string formula specifying the report to launch, parameter values, and even User ID and Password.

The general structure of the String formula (no naming restrictions) is:

- 1) **DLV_Run:-v** (launch specified report in a **new Preview Only window**) or
DLV_Run:-s (launch specified report in a **new full DataLink Viewer window**) or
DLV_Run_Here:-s (launch the second report **within the current Preview Tab**)
- 2) the **report path and file name**
- 3) **Optional command line arguments** (parameter values, login information, etc.) as discussed in the "Launch Reports from Command Lines" section of this user manual.

Notes:

- The **User ID & Password values from the current report are automatically passed to the new report**. Specify these arguments only if you need to override these values.
- **-v** should be the **preferred option** when you launch a report into a new window because it removes the Select Report tab from the new window, simplifying what the user sees. This also increases the window space dedicated to showing the report.

Here are some examples of such String formulas:

```
// invoke a second report and load it into the Current Preview Tab
```

```
'DLV_Run_Here:-s "c:\directory\subdirectory\Report.rpt'
```

```
// launch into a Preview Only (-v) window and specify some parameter values:
```

```
"DLV_Run:-v ""c:\CR\My_Report.rpt"" ""Parm1:" + {Cust.CUST_N} +  
"" ""Parm2:" + {Prod.PROD_Code} +"""
```

```
// full DLV window (-s) and specify both login as well as Parameter information:
```

```
'DLV_Run:-s "Some_Report.rpt" "user_id:dba" "password:sql" "Parm1:1997"
```

Note:

- **No Text Wrapping Allowed**, so use very small font, and **to hide the formula you should use font colors, not the Suppress attribute**.
- **Position the formula box and expand its borders to cover the intended clickable area**.
- When you need to embed double quotes into the result of a string formula, you must "escape" the double quotes with another double quote. Here are some variations on that theme:
 - ...+ "" + add a double quote.
 - ""..."" start the resulting string with a double quote
 - ""..."" end the resulting string with a double quote
 - ""...""..."" insert a double quote into the middle of the resulting string

Create a User Interface for Selecting and Launching Reports

The ability described above (launching one report from another) allows you to very easily create a user interface that lets your users select and launch reports. The key steps are:

1. Create a REPORT Table with one record for each report. Typical columns would include: a) the path and report file name, b) a description of the report, c) report subject (Profit, Product Returns, Orders), d) report frequency (Daily, Weekly, Monthly), etc.
2. Create a “Report Selection” report listing these available reports with a formula in the detail section for launching the selected report. Note: you can design the main report with Parameters for listing only reports of certain subjects or certain frequencies. Of course, the REPORT table must store these classifications.

Let the user run the Report Selection report and launch the selected report by double-clicking it.

Embed Input from The User in the Command Line Call for Another Report

When launching another report by double-clicking a report section (as described above) you may wish to prompt the user for some input and embed that input into the command line. One scenario is a case where you wish to prompt the user to provide the value for a parameter. Another scenario may be a case where you wish to print labels for the product you are double-clicking but you want the user to be prompted for the number of copies to print.

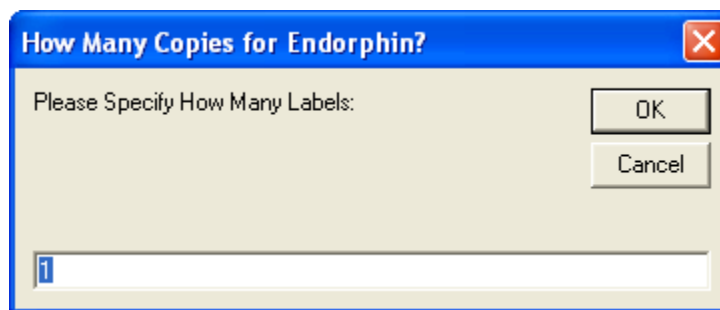
You can achieve this functionality by embedding within the command line a call to an InputBox function. DataLink Viewer would then replace the InputBox Call with the input from the user.

For example, the following formula would prompt the user for number of copies and then print the Product.rpt report for the double-clicked Product Name to the default printer with the number of copies specified by the user:

```
"DLV_Run:-v ""C:\temp\Product.rpt"" ""Parm1:" +  
{Product.Product Name} + """" +  
" ""Printer:Default"" +  
" ""Print_Copies:#{InputBox||Please Specify How Many Labels:" & "||" &  
"How Many Copies for " & {Product.Product Name} & "?" &  
"||" & "1}#"""
```

The bold text is the InputBox call. This call must be enclosed by #{... }# and has 4 parts separated by "||":

1. **InputBox** (a keyword indicating this is a request for an InputBox dialog)
2. The **Prompt** (the text shown within the InputBox dialog. If you need to specify line breaks within the prompt text, be sure to use "**vbCrLf**" (case sensitive), rather than Chr(10).
3. The **Title** (the text shown as the title of the InputBox dialog.
4. The **Default** value (**1** in the example above).



Notes:

1. On the report itself, the formula **must be displayed on a single line**, so use small font
2. to hide the formula you should use font colors, not the Suppress attribute

Launch another Application and Pass Parameters to It

DataLink Viewer lets you launch any other application and pass parameters to it by double-clicking a report section that contains a formula that statically or dynamically results in a string with the following structure:

'EXE_Run:Any_Program.exe:command line arguments'

Note:

- Note: **No Text Wrapping Allowed**, so use very small font, and **to hide the formula you should use font colors, not the Suppress attribute**.
- When you need to embed double quotes into the result of a string formula, you must "escape" the double quotes with another double quote. Here are some variations on that theme:
 - ...+ "" + add a double quote.
 - ""... " start the resulting string with a double quote
 - "..."" end the resulting string with a double quote
 - "...""..." insert a double quote into the middle of the resulting string

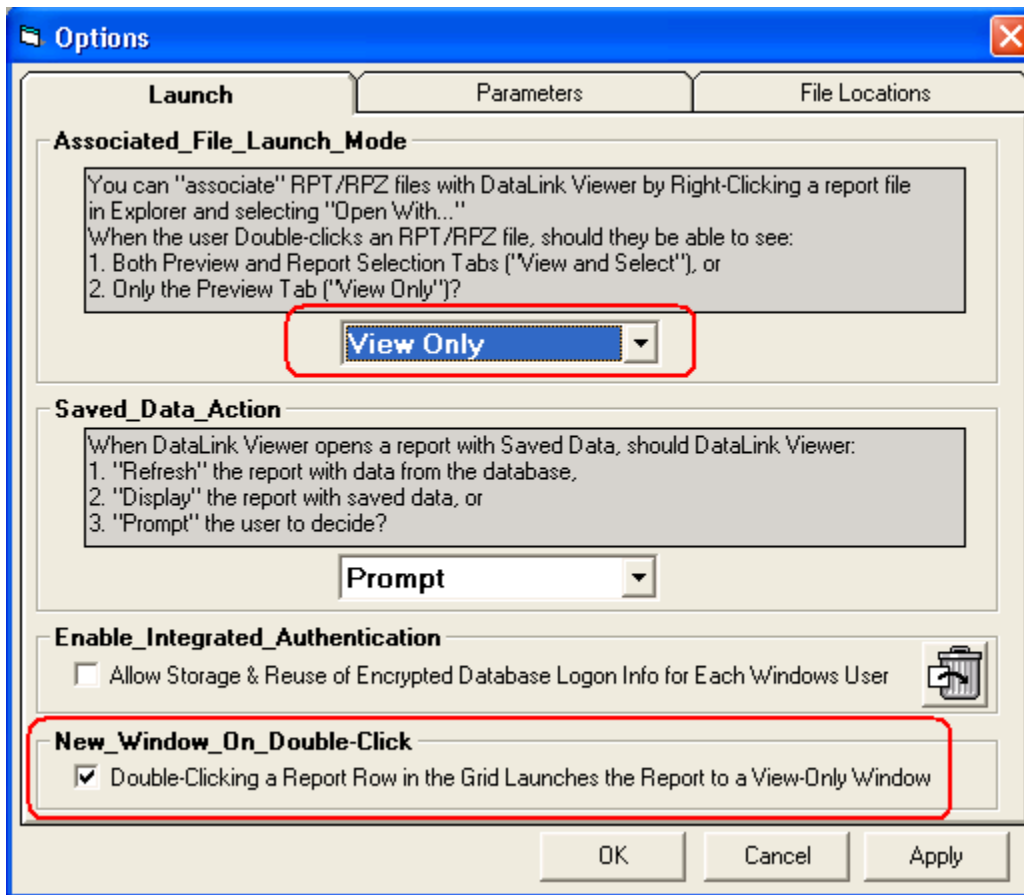
Launching Reports in a New Window

By default, the initial DataLink Viewer (DLV) window displays two tabs: 'Select Reports' and 'Preview.' You select a report from the first tab, and view it in the second tab. However, in many cases you may wish to **launch a report from the initial window into a new preview only window**. You may also wish to **remove the Preview tab from the initial window**. This section describes the various options for achieving this:

As described in the sections dealing with the command line api of DLV, you can launch reports in a Preview Only window by using `-v` instead of `-s` within the command line. However, this section deals with user interaction options for achieving the same thing. There are several ways to achieve this.

NEW WINDOW ON FILE LAUNCH AND/OR ON DOUBLE CLICK

The Launch tab in the DLV Options dialog allows you to direct DLV to launch reports into a View Only window when double-clicking an rpt file in File Explorer or when double-clicking a report row in the Select Report tab:



RIGHT-CLICK THE GRID AND SELECT PREVIEW REPORT (NEW WINDOW)

When you right-click a report row in the Select Report tab, a popup menu provides you an option of **Preview Report (new window)**. If you click on that option, the report would launch in a new window.

REMOVE THE PREVIEW TAB FROM THE INITIAL WINDOW

If you wish to always use the initial DLV window to launch reports into new preview only windows, you may want to completely remove the Preview tab from the initial DLV window. You can achieve that by setting the following options in the **DataLink_Viewer.ini** file:

[Options]
Show_Preview_Tab=FALSE

Note: this option takes effect only if you also use the Options dialog to turn on the option for Double-Clicking a Report Row in the Grid Launches the Report to a View Only Window.

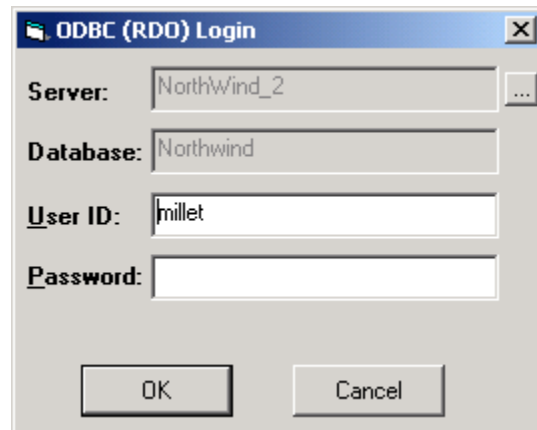
When these two options are set, DLV changes several things:

1. The initial DLV window shows only a Select Report tab
2. The right-click menu option of **Preview Report** is not shown. Instead, only the **Preview Report (new window)** option is visible.
3. The right-click menu option of **Preview Report (force login)** launches the report to a new window instead of to the Preview tab (which is not available).

The end result is that the initial DLV window becomes just a launch pad to report previews.

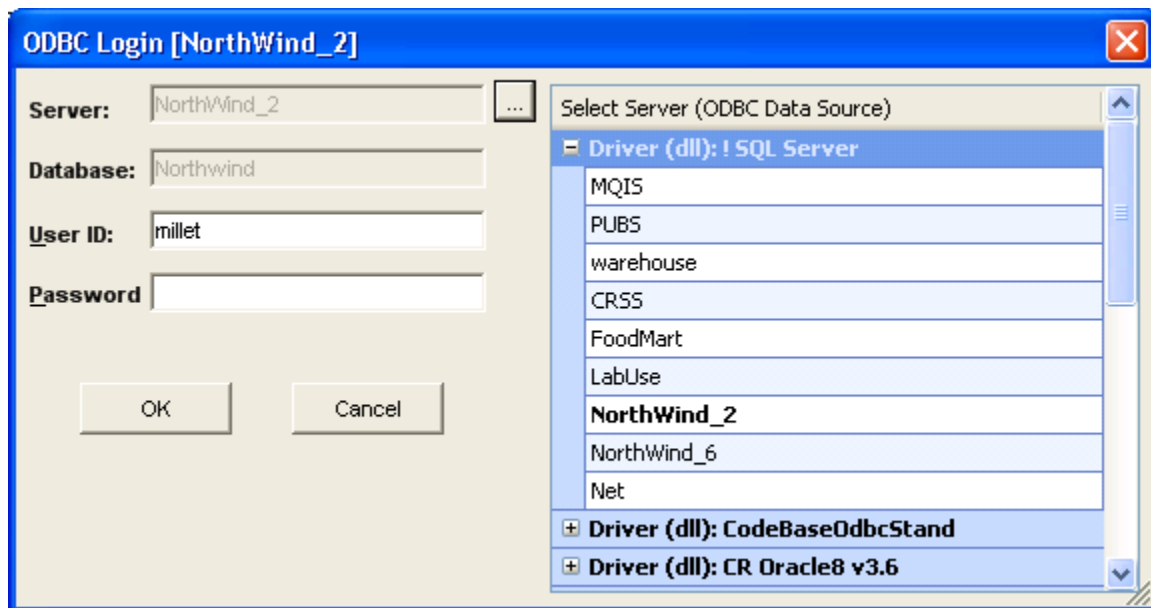
Database Choice Functionality

DataLink Viewer provides login dialogs to support any number of secure data sources in the main report as well as the subreports. Here is what the login dialog looks like:



Select Alternative ODBC Data Sources for the Same Report

When running reports that use ODBC data sources, you can **select which ODBC data source should be used**. Clicking a button to the right of the Server name, expands the display to include a listing of all available ODBC data sources (grouped by ODBC driver type). The original ODBC data source is initially selected and the driver group it belongs to is expanded (and prefixed with a "!"):



Restricting the List of ODBC DSN Choices

The default ODBC DSN used by a report may belong to what you may consider to be a group of alternative ODBC DSNs. You may wish to restrict the list of ODBC DSNs options shown for such a report to only those that belong to that group. This can be accomplished by specifying an **ODBC_Groups** entry under a **[Login_Window]** section in the **DataLink_Viewer.ini** file.

Here is an example of such an entry:

```
[Login_Window]
ODBC_Groups={ Xtreme 9 Xtreme 11 } { Northwind_2 NorthWind_6 Xtreme 9 }
```

Each ODBC Group is enclosed in { } brackets.

A single space separates the groups.

Each ODBC DSN is enclosed in a | delimiter (including at the start and end of the group).

report that by default uses an ODBC DSN that is NOT in any of the specified groups will experience no restrictions in the choice of alternative ODBC DSNs.

A report that by default uses an ODBC DSN that is in only one of the specified groups will be restricted to a choice among the DSNs of that group.

A report that by default uses an ODBC DSN that is in more than one of the specified groups will be restricted to a choice among the combined list of DSNs from all these matching groups.

Using Command Line Arguments to Specify the ODBC DSN

In some cases, you may want to use the same report to connect to different data sources, such as a **testing** or **production** server. While each report stores connection properties for only one default ODBC Data Source Name (DSN), DataLink Viewer allows you to use command line arguments to specify a different ODBC DSN.

Note: You can combine this functionality with the **user_id** & **password** command line arguments described in a previous section.

The command line argument structure is as follows:

... "**ODBC_DSN:Data Source Name**"

or

... "**ODBC_DSN_From_To:Old_DSN1>>New_DSN1||Old_DSN2>>New_DSN2**"

The **ODBC_DSN** argument overrides all ODBC DSNs used in the report by the new DSN

The **ODBC_DSN_From_To** argument overrides only for tables that use the old DSN.

Note: **ODBC_DSN_From_To** support **multiple pairs** separated by “||” as shown in the example above. This addresses scenarios where a report uses multiple ODBC DSNs (e.g. when subreports use different DSNs).

Overriding the Database Specified in the Report or ODBC DSN

For ODBC data sources, you can enter a database name into the login dialog if you wish to override the database specified in the ODBC data source or in the report itself. This is useful for situations where the same database (e.g., MS SQL Server) contains multiple databases each with the same table structure. If the number of such databases is large, creating a dedicated ODBC DSN for each and using the select ODBC DSN functionality may be too tedious. Instead, you can directly type in the database name in the login dialog.

To enable database name input into the database field in the login dialog, you need to set the **Override_ODBC_DSN_Database** entry under the [Options] section in the DataLink_Viewer.ini file to TRUE, like this:

```
[Options]  
Override_ODBC_DSN_Database=TRUE
```

Note that if that option is set to TRUE, the database specified in the ODBC DSN can no longer override the database specified in the report (if the user doesn't type in a Database, the specified Database remains the one used in the report).

Note: This functionality is not available in the Crystal 8.5 version of DataLink Viewer.

Overriding the Table Location

This functionality was added for a developer who needed to call DataLink Viewer from his application via a command line. The data source was Pervasive via ODBC and a command line argument was needed to control what year archive is used for the report. The database contained a table for 2010 (**GL-10JRL**) as well as a similar table for 2011 (**GL-11JRL**).

The report was already designed to reference the **GL-10JRL** table using a generic alias of **GL_JRNL**.

The command line argument to allow overriding the table used for the report looks like this:

```
... "Table_From_To:GL-10JRL>>GL-11JRL| |AR-10JRL>>AR-11JRL"
```

Within each pair of From/To directives, the 'From' location is separated by a '>>' from the 'To' location. The pairs are separated by a "||" from each other.

Notes:

1. If any of the *From* tables is not found in the report, the report loading stops with a message indicating which From tables were unmatched with report tables.
2. This functionality is available only in DataLink Viewer 2008 and 2011.

Overriding the XML File Location

This functionality was added for a customer who needed to call DataLink Viewer from his application via a command line. The data source for the report was an XML file (ADO.NET XML connection), but on each call to DataLink Viewer, instead of using the original XML file as a data source, a different XML file (name and location) may be used.

The command line argument to allow overriding the XML file path looks like this:

```
... "XML_Path_From_To:C:\custprog\PICKLIST.xml>>C:\temp\B139.xml"
```

If you need more than one pair of **From/To** directives, separate them with a '||' delimiter. Within each pair of **From/To** directives, the 'From' location is separated by a '>>' from the 'To' location.

Notes:

1. If any of the **From** paths is not found in the report, the report loading stops with a message indicating which From paths were unmatched.
2. This functionality is available only in DataLink 2011.

Stripping Table Qualifiers when Connecting to a Different Database

The SQL statement generated by Crystal frequently contains not only table names, but also the database name used at the time the report was designed (Database.Table or Database.Owner.Table). Such table qualifiers can frustrate attempts to run the report against a different database.

DataLink Viewer supports selection of a new ODBC data source even when the database name is different from that used in the original ODBC DSN. To ensure this works even when the table names in your report are fully qualified, set the following line under the **[Options]** section in **DataLink_Viewer.ini** to True:

```
Strip_Table_Qualifiers=True
```

Overriding the Server in Native Oracle Connection

When a report uses a **native connection** to Oracle, you can **edit the Server name** in the login dialog and run the report against a different server (rather than the one the report was designed against).

Alternatively, if you are launching a report from a command line, you can override the Oracle server name by using the "Oracle_Server:" command line argument. For example:

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v  
"C:\temp\test.rpt" "user_id:dba" "password:sql" "Oracle_Server:Server2"
```

Note: This functionality is not available in the Crystal 8.5 version of DataLink Viewer.

Selecting an Alternative SQL Server – OLE DB Data Source

... "**Connect_To_SQLOLEDB:DataSource>>InitialCatalog>>Integrated_Auth**"

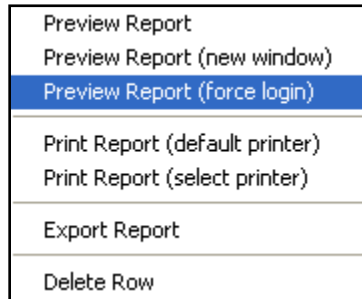
The parameters (after the “:”) are separated by a “>>” and are as follows:

1. **DataSource** is the **Server Name** the report should connect to.
2. **InitialCatalog** is the **Database Name** within the given server.
3. **Integrated_Auth** is a **True** or **False** argument indicating if Microsoft SQL Server Integrated Authentication should be used (if TRUE, user_id & password are ignored).

Note: This functionality is not available in the Crystal 8.5 version of DataLink Viewer.

Forced Login

Right-clicking the report list provides a **Preview Report (force login)** popup menu option for forcing a login window before previewing the report:



This is useful when:

- a previous report connected to one ODBC data source and you wish to run the same (or another) report against another ODBC data source.
- you wish to login to the same data source under a different user id, without restarting DataLink Viewer.
- You wish to avoid the default login information provided by Integrated Authentication (without bothering to go into the Options dialog and turning Integrated Authentication off)

Note: for some ODBC data sources (Crystal Commands), there are some scenarios where the only way to avoid connecting to a previously opened connection is to close and reopen DataLink Viewer.

Selecting Folder Location for FoxPro DBF Files (MasterBuilder)

DataLink Viewer 2011 supports dynamic selection of data folders for reports using the Visual FoxPro ODBC driver for a File DSN (using “Free Tables”, which are dbf files under a given folder). IMPORTANT NOTES:

1. The File DSN must be present (and user should have modify permissions) at:
**C:\Program Files\Common Files\ODBC\Data Sources\
or, on a 64-bit machine, C:\Program Files (x86)\Common Files\ODBC\Data Sources\
(but not in both locations)**
2. The folder choice actually changes the File DSN, so you may wish to create a dedicated File DSN just for your Crystal Reports.

This functionality was originally developed to support *Intuit Master Builder* company reports where each company’s data is located in a different folder on the same PC.

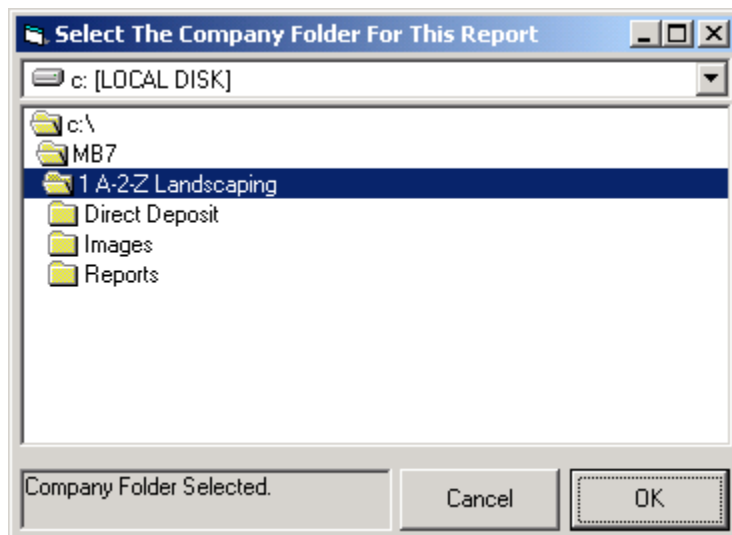
Two sections in the DataLink_Viewer.ini file control this functionality:

```
-----  
[Folder_Selection_DSNs]  
Visual_FoxPro_DBF=||MB7 Data.dsn||MB Data.dsn||
```

```
[MB7 Data.dsn]  
Folder_Selection_Must_Contain=company.dbf  
Enable_Folder_Selection=TRUE  
-----
```

The 1st section specifies which File DSNs call for this new functionality. The File DSNs are specified exactly as they are named in the report itself. They must be enclosed in “||” as delimiters (even for the first, last, or only entry)

The 2nd section specifies, in the case of Master Builder data, that users should be allowed to select only folders that contain a **company.dbf** file. That section also allows users to Enable or Disable dynamic folder selection.



Changing Folder Location for Access/Excel/Pervasive (ddf) Files

If your report uses the native connection to MS Access or Excel files, you can control the location of the database files using the following section in DataLink_Viewer.ini

```
-----  
[Database_Path_Selection]  
Paths = C:\Old\xtreme.mdb>>C:\New\xtreme.mdb|C:\a\test.mdb>>?  
// Or, in the case of Pervasive ddf files:  
// Paths = E:\Jobtrack\FILE.DDF>>?  
-----
```

As demonstrated above, the Paths entry may contain multiple pairs of old>>new paths. Each pair specifies the **old path** followed by a ">>" separator, followed by the new path. The pairs are separated by "|"

If the new path is blank, or if it contains just a question mark, DataLink Viewer will prompt the user to select a new path when a report using the old path is first used.

If the new path has one question mark (?) followed by a valid path, that path will be the default location in the dialog asking the user to select a path.

If instead of a single question mark, the new path starts with a double question mark (??), DataLink Viewer will always prompt the user to select a path each time a report using the old path runs. The user choice will be added after the ?? and will become the new default value in the path selection dialog.

If the report uses a database file that can't be found on the machine, DataLink Viewer prompts the user to select a valid location, creates the [Database_Path_Selection] section (if it doesn't already exist) and creates/adds the pair information to the Paths entry.

These options are designed to address typical deployment scenarios when a report developer sells reports to users who may have their data source location at a different folder than the one used when the reports were developed.

USING COMMAND LINE ARGUMENT TO AVOID PATH PROMPT

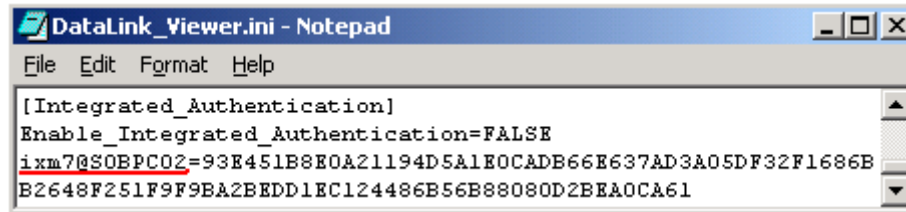
By default, pairs with two question marks before the new default path (old_path>>??new_path) cause a prompt each time the old path is encountered in a report. You may wish to avoid such repeated prompts when launching one report from within another report or when scheduling printing. The following command line argument will cause DataLink Viewer to ignore the two question marks and simply use the new_path.

... "DB_Path_Use_Default:TRUE"

Integrated Authentication

Integrated Authentication ("Remember Me")

DataLink Viewer allows users to **avoid repeated login prompts to databases**. The database login information is stored, highly encrypted, inside **DataLink_Viewer.ini** as shown below:



```
DataLink_Viewer.ini - Notepad
File Edit Format Help
[Integrated_Authentication]
Enable_Integrated_Authentication=FALSE
ixm7@SOBPC02=93E451B8E0A21194D5A1E0CADB66E637AD3A05DF32F1686B
B2648F251F9F9BA2BEDD1EC124486B56B88080D2BEA0CA61
```

The security of login information is maintained not only by storing it in **encrypted** format, but also by storing it with an internal identification of which **Windows User & PC** this login information belongs to. **DataLink Viewer uses this encrypted database login information only after checking that the same Windows user is running from the same PC.** In other words, users cannot break the login security by attempting to copy and paste the encrypted information to their own ini file entry).

For example, in the example shown above integrated authentication has been enabled. This can be done via a checkbox in the Option dialog (Launch Tab). The user (**ixm7**) then logged in to a secure database by providing a **database user id & password**. The user turned on the **"Remember Me"** option (visible only when integrated authentication is enabled):



The information was then saved for the **ixm7** user, running on the **SOBPC02** PC as shown in the ini file above. **From that point on, the same user (ixm7), once logged to the same PC (SOBPC02), doesn't need to manually login to the same data source.**

The same user can turn on the "Remember Me" option for **unlimited number of secure data sources** and the information for all of them would be maintained inside the encrypted entry. The Options dialog allows each user to delete their own integrated authentication information by clicking a button.

Shared Machine Authentication

This section describes how one user can elect to share their integrated authentication information with any other user who successfully logged in to the same machine.

Step 1: First, add the entry in bold to the DataLink_Viewer.ini file:

```
-----  
[Integrated_Authentication]  
Enable_Integrated_Authentication=TRUE  
ixm7@SOBPC02=18CB9CAE3D8BF3484000123301DD155638EAD4AD6B4D622C04DF125B7404BCBC4717A5DA2FB CD94314A7CE63BBF7357E  
Enable_Shared_Machine_Authentication=TRUE  
-----
```

Step 2: Then, as the "key" user who will share integrated authentication, run a report and get to a login dialog. Note: if you already have integrated authentication for yourself, use Forced Login. Alternatively, discard your integrated authentication (using the Trash Can button in the Options dialog). Make sure you turn on the Remember Me option in the login dialog, just like setting up integrated authentication for yourself.

Step 3: Because of the **Enable_Shared_Machine_Authentication=TRUE** you would get a dialog asking you if you wish to share your integrated authentication functionality with other users who logged in to the same machine. This ensures a user can't be tricked into sharing integrated information without their knowledge and expressed consent. Click YES.

If you then open the DataLink_Viewer.ini file you would notice this process generated a new entry (in bold):

```
-----  
[Integrated_Authentication]  
Enable_Integrated_Authentication=TRUE  
ixm7@SOBPC02=18CB9CAE3D8BF3484000123301DD155638EAD4AD6B4D622C04DF125B7404BCBC4717A5DA2FB CD94314A7CE63BBF7357E  
Enable_Shared_Machine_Authentication=TRUE  
Shared_Machine_Authentication=C8B6CD373D5D5ADDC6FE9F379521C7318FA297CB595B992C  
-----
```

That entry, in my particular case, points to the **ixm7@SOBPC02** integrated authentication entry.

Notes:

1. The **Enable_Integrated_Authentication** (True or False) setting is **always taken from the "user" ini file**. This allows the user to turn it off if they wish. For example, they may wish to switch between ODBC Data Sources by using the special option in the login dialog.
2. The **Enable_Shared_Machine_Authentication** and **Shared_Machine_Authentication** information is **always taken from the "master" ini file**. This allows an administrator to enable/disable and change the shared machine authentication for all users in one central file.
3. The regular integrated authentication information (for a **user@machine** entry that matches the logged in user id and the machine id) is always taken from the user ini file, if it exists.

4. **There can be only one shared machine authentication entry.** If you need to change to another administrator, delete the **Shared_Machine_Authentication** line and let the new administrator go through the Remember Me and dialog step.

5. If the administrator adds more data sources and login information to their integrated authentication information, all the machine users would have access to the new data sources. This is because the entry is really a "pointer" to whatever that administrator has accumulated in their entry (ixm7@SOBPC02 in my case).

Shared Secret Password with Windows User IDs

DataLink Viewer has special functionality allowing an administrator to set/change a secret global password for all users who will then be authenticated to the database using their own windows user id and the secret global password.

If you are in the rare situation where you need to use this functionality, contact Millet Software and, if your use scenario matches this functionality, you will receive detailed instructions.

Protect Report Designs with rpz Files



Compress & Encrypt Rpt to Rpz

The 'Compress & Encrypt Rpt to Rpz button' (on the 1st Tab of DataLink Viewer) allows you to select an rpt file and convert it to **rpz** file.



The resulting **rpz** file is a compressed and encrypted version of the rpt file that is recognizable only by DataLink Viewer & Visual CUT. Your users can run the resulting **rpz** files in DataLink Viewer or Visual CUT, but cannot view or modify them in Crystal.

This allows developers to protect and hide their reports designs (either as an intellectual property issue or as a tech support issue). They can simply keep the rpt files and distribute only the **rpz** files to their users. Note: .rpz files should not be renamed.

Within DataLink Viewer & Visual CUT, **rpz** files behave just like rpt files except that, in order to protect the report design, exporting **rpz** files to "rpt" format is not possible.

Note: DataLink Viewer provides a special export format of "**Protected Report (*.rpz)**". This allows users to export the report with saved data to another .rpz file.

Make_Rpz Command Line Argument

If you need to automate the conversion of rpt files to rpz files using scheduled or automated processes, DataLink Viewer provides a command line API.

Your command line call should look like this (all in one line):

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe"
```

```
"Make_RPZ:c:\test\*.rpt>c:\MyFolder>120"
```

After the Make_RPZ key word and the colon, the arguments are separated by ">" as a delimiter. The three arguments are:

1. The path to the rpt file(s) to be converted.
 - you may specify wild cards (as demonstrated in the example above)
 - you may specify multiple paths separated by a semi-colon. For example:
"Make_RPZ:c:\rptFolder1*.rpt;c:\rptFolder2*.rpt>c:\rpzFolder>120"
2. The target folder where the rpz files should be deposited
3. The maximum age, in minutes, of the rpt file in order to be included. Older files would be skipped. This allows the command line to be placed in a batch file and scheduled while avoiding the conversion of older files that have already been converted. Use a value of zero to convert all files regardless of age.

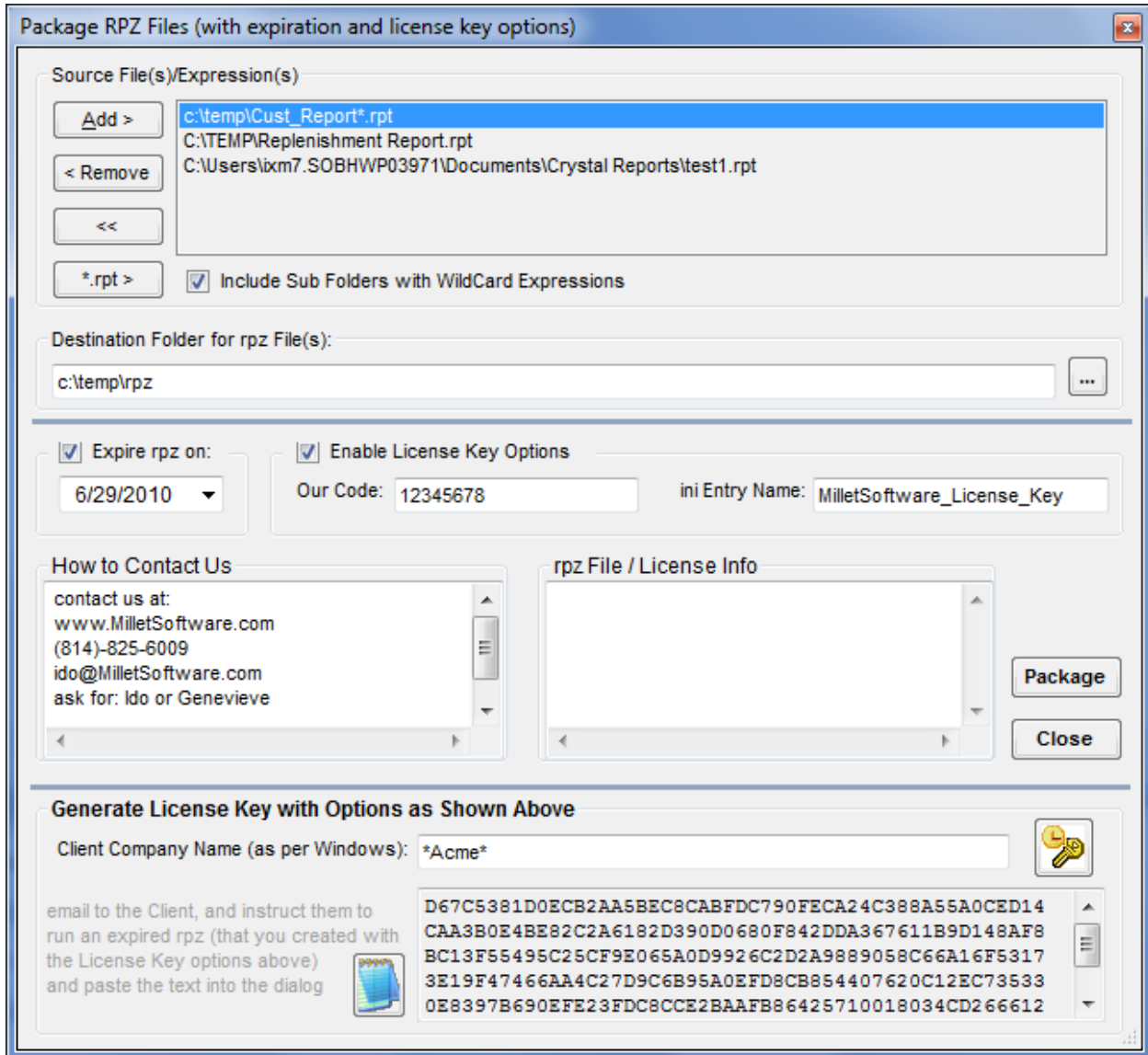
Create rpz Files with Expiration and License Keys

You can apply expiration date and/or license keys to rpz files by setting the following entry in the DataLink_Viewer.ini file:

[Options]

Enable_Advanced_RPZ_Options=True

When that option is enabled, the dialog you get when clicking on the ‘Compress & Encrypt Rpt to Rpz’ button looks like this:



Please contact Millet Software for more detail about the various options and the use scenarios supported by this dialog.

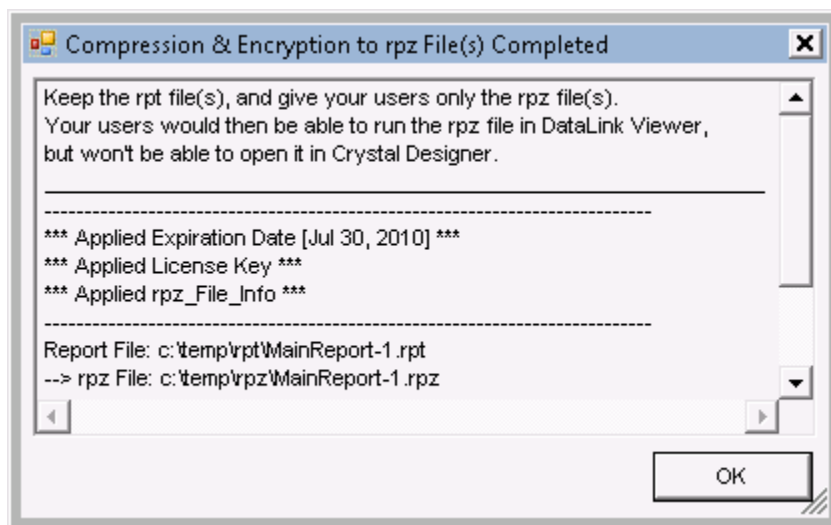
MAKE_RPZ2 COMMAND LINE ARGUMENT

If you need to automate the conversion of rpt files to rpz files with expiration dates and license keys, your command line call should look like this (all in one line):

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe"  
"Make_RPZ2:c:\test\*.rpt>>>  
c:\MyFolder>>>False>>>2221440>>>rpz file info line 1/////and line 2>>>30/07/2010>>>  
MilletSoftware_License_Key1>>>12345678>>>Call us at (814) 825-6008/////ido@MilletSoftware.com>>>  
True"
```

After the Make_RPZ2 key word and the colon, the arguments are separated by ">>>" as a delimiter. The 8 arguments are:

1. The **path to the rpt file(s) to be converted**.
 - you may specify wild cards (as demonstrated in the example above)
 - you may specify multiple paths separated by a semi-colon.
2. The **target folder** where the rpz files should be deposited
3. **Include subfolders** when wildcard expressions are used for source files (True/False)?
4. The **maximum age**, in minutes, of the rpt file(s) to be included. Older files would be skipped. This allows the command line to be placed in a batch file and scheduled while avoiding the conversion of older files that have already been converted.
Use a value of zero to convert all files regardless of age.
5. **Text information to embed in the rpz file**. Separate lines with '/////'. This information can be accessed via a report parameter named DLV_Rpz_File_Info
6. **Rpz Expiration Date** (dd/mm/yyyy). The rpz file will not run beyond that date, unless a license key with a later expiration date is added to the machine.
7. **Contact Information** to embed in the rpz file. This information is provided to the user when trying to run an expired rpz or an rpz requiring a new license key.
8. **Show Conversion Results Dialog** (True/False). This controls whether the process will be "quiet" (typical for a scheduled process) or result in a dialog showing what files were converted and what options were applied:



Monitoring DataLink Viewer Use

DataLink Viewer allows you to monitor the use and performance of reports across the organization via logging to an ODBC table or to a simple text file. Logging to ODBC is a more secure and powerful approach and is described first.

Record Processing to an ODBC Database

To log processing to an ODBC database, you must create a table called DLV_Log in the target database. Email to me if you'd like to receive a sample MS Access database with the required DLV_Log table.

Below are example of data structures for MS Access and MS SQL Server. However, those are just examples. You can use such a table in any other ODBC aware database (Oracle, DB2, Sybase, etc.).

TYPICAL USE

The DLV log table can be used to:

1. Monitor report use
2. Monitor report performance (using the start & end time information)
3. Monitor what employees access/print/export what data. This can be useful for addressing data privacy and protection concerns and requirements (e.g., HIPAA)

If there is a need to monitor this log for exception situations, you can use Visual CUT (another Millet Software tool) to schedule exception reports and email alerts against this table.

MS ACCESS DATA STRUCTURE

Here is the table structure required for this table, when implemented under MS Access (along with comments explaining what information is recorded):

Field Name	Data Type	Description
LogN	AutoNumber	Surrogate Key
Rpt_Path	Text	Path to the rpt file
Rpt_Name	Text	Name of the rpt file
Proc_Start_Local	Text	Processing Start DateTime - String representation. For example: 10/18/2008 3:48:02 PM
Proc_End_Local	Text	Processing End DateTime - String representation. For example: 10/18/2008 3:48:04 PM
Proc_Start_GMT	Text	Processing Start DateTime in Greenwich Mean Time (GMT, also called UTC or Zulu Time).
Proc_End_GMT	Text	Processing End DateTime in Greenwich Mean Time (GMT, also called UTC or Zulu Time)
Windows_User_ID	Text	The Windows User ID running the process
Machine_ID	Text	The Windows machine running the process
Parameter_Values	Memo	Param_Name=Param_Value(s) [subreport Name]->Param_Name=Param_Value(s)
Connection_Properties	Memo	Delimited List of Connection Property Names & their Values (not including passwords).
Report_SQL	Memo	The SQL Query for the main report
Live_Data_1_0	Number	1 if user retrieved data from Database . 0 if saved data in the report was used.
Records_Read_N	Number	Number of Records Read into the Main_Report
Export_Activity_1_0	Number	1 (Yes) or 0 (False)
Print_Activity_1_0	Number	1 (Yes) or 0 (False)

SQL SERVER DATA STRUCTURE

Here is a script for creating the table in Microsoft **SQL Server**:

```
CREATE TABLE [dbo].[DLV_Log] (  
[LogN] [int] IDENTITY (1, 1) NOT FOR REPLICATION NOT NULL ,  
[Rpt_Path] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Rpt_Name] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Proc_Start_Local] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Proc_End_Local] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Proc_Start_GMT] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Proc_End_GMT] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Windows_User_ID] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Machine_ID] [nvarchar] (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Parameter_Values] [nvarchar] (40000) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Connection_Properties] [nvarchar] (40000) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Report_SQL] [nvarchar] (40000) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Live_Data_1_0] [int] NULL ,  
[Records_Read_N] [int] NULL ,  
[Export_Activity_1_0] [int] NULL ,  
[Print_Activity_1_0] [int] NULL  
) ON [PRIMARY]  
GO
```

Note: for recent versions of SQL Server, replace **[nvarchar] (40000)** with **[nvarchar] (MAX)**

ORACLE DATA STRUCTURE

The script below, contributed by Jonathan Klobucnik from A.H. Belo, also includes a definition of a Sequence and a Trigger to handle the Auto-Numbering requirement:

```
CREATE TABLE DLV_LOG
(
  LOGN NUMBER(*, 0) NOT NULL , RPT_PATH NVARCHAR2(255) , RPT_NAME NVARCHAR2(255)
, PROC_START_LOCAL NVARCHAR2(255) , PROC_END_LOCAL NVARCHAR2(255)
, PROC_START_GMT NVARCHAR2(255) , PROC_END_GMT NVARCHAR2(255)
, WINDOWS_USER_ID NVARCHAR2(255) , MACHINE_ID NVARCHAR2(255) , PARAMETER_VALUES CLOB
, CONNECTION_PROPERTIES CLOB , REPORT_SQL CLOB , LIVE_DATA_1_0 NUMBER(*, 0)
, RECORDS_READ_N NUMBER(*, 0) , EXPORT_ACTIVITY_1_0 NUMBER(*, 0)
, PRINT_ACTIVITY_1_0 NUMBER(*, 0) , CONSTRAINT DLV_LOG_PK PRIMARY KEY
(
  LOGN
)
ENABLE
)
LOGGING
PCTFREE 10
INTRANS 1
STORAGE
(
  BUFFER_POOL DEFAULT
)
LOB (PARAMETER_VALUES) STORE AS
(
  ENABLE STORAGE IN ROW
  CHUNK 8192
  RETENTION
  NOCACHE
  LOGGING
)
LOB (CONNECTION_PROPERTIES) STORE AS
(
  ENABLE STORAGE IN ROW
  CHUNK 8192
  RETENTION
  NOCACHE
  LOGGING
)
LOB (REPORT_SQL) STORE AS
(
  ENABLE STORAGE IN ROW
  CHUNK 8192
  RETENTION
  NOCACHE
  LOGGING
);
/
/*Sequence to allow the trigger to "auto number" the primary key*/
CREATE SEQUENCE DLV_LOG_SEQ;
/
/*Trigger to automatically generate the primary key values*/
CREATE OR REPLACE TRIGGER DLV_LOG_INS
BEFORE INSERT ON DLV_LOG
FOR EACH ROW
WHEN (new.LogN IS NULL)
BEGIN
  SELECT DLV_LOG_SEQ.NEXTVAL
  INTO :new.LogN
  FROM dual;
END;
```

“ORACLE” MODE (BIND VARIABLES)

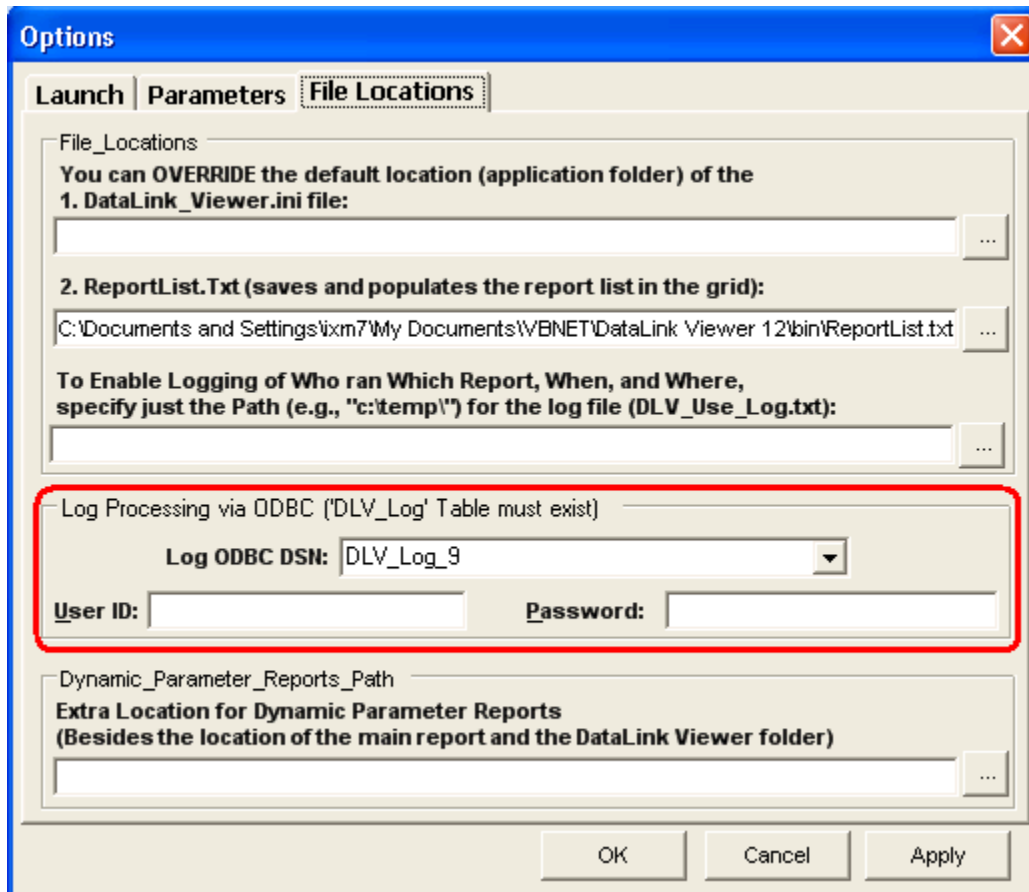
If you wish to log processing to an Oracle database, you should add the following entry to the [Options] section of DataLink_Viewer.ini:

Log_ODBC_Type=Oracle

That option sends Insert SQL statements using bind variables, an approach that avoids a 4K column size limitation.

HOW TO START LOGGING?

The following DataLink Viewer Options dialog allows you to specify the ODBC Data Source Name (DSN) where the DLV table resides and the User ID & Password (stored encrypted), if the data source requires a login.



Record Report Use in a Text Log File

If you use the Options dialog to specify a folder location for a **DLV_Use_Log.txt** DLV creates that file (if it doesn't already exist) and appends to it use statistics after each report viewing. This log file contains one row with column headers followed by one row for each report viewing event. The columns are:

- 1) Report Path
- 2) Report Name
- 3) User_ID
- 4) PC
- 5) Start
- 6) Finish

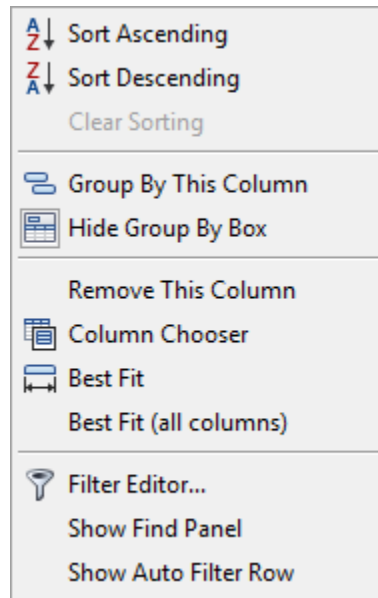
You can use that file (for example, using Crystal with a Text ODBC driver) to analyze what reports are popular, who runs what reports, and what reports take too long to run.

Settings & Options

Customizing the Grid Layout

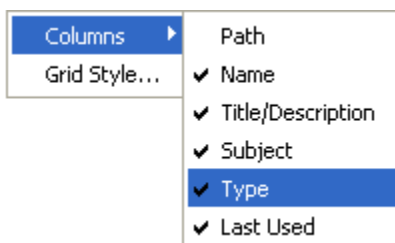
By dragging, clicking, or right-clicking the grid column headers you can apply various options such as grouping the report by any column(s), sorting the grid, hiding/showing columns, etc.

Here is the menu you get when you **right-click a column header**:



Most of these options are quite intuitive. The ‘Group By Box’ shows or hides the area at the top of the screen that allows you to control how the Grid is grouped by dragging & Dropping column headers to/from that area.

Here is the menu you get when you **right-click the top-left corner of the grid**:

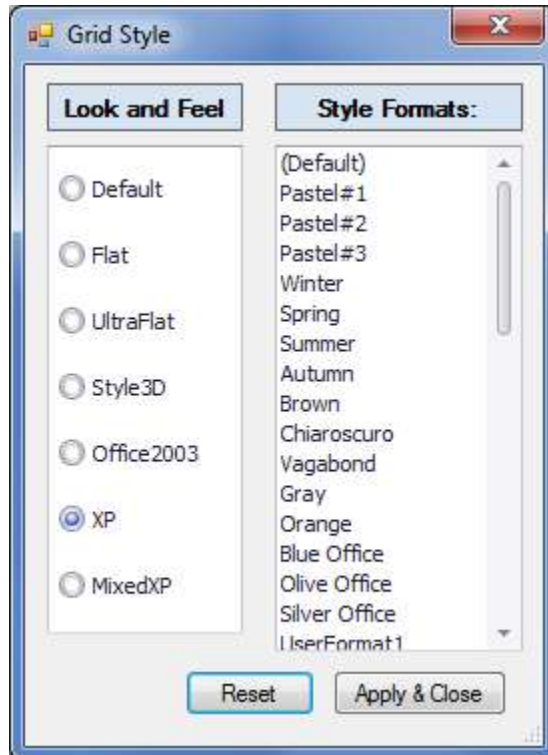


As shown above, the Columns menu cascades to the list of columns and allows you to directly set column visibility. You can do the same via a slightly longer process using the “Column Chooser” option in the previously shown menu.

Customizing the Grid Style

The “Grid Style” option in the menu you invoke by **right-clicking the top-left corner of the grid** bring up a dialog that allows you to customize the look & feel of the grid. Any choice you make in the Grid Style dialog shown below is immediately reflected in the style of the grid.

The style of the grid is maintained in **ReportList.grd**. You can always revert back to the default style of the application by closing DataLink Viewer, Deleting that file, and starting DataLink again.



Disabling Report Preview Buttons

In some scenarios you may wish to remove the Print, Export, Select Expert, or Search Expert buttons from the preview window of reports. This is useful, for example, in cases where your application launches reports in DataLink Viewer via a command line and you wish to restrict some users from printing and/or exporting the report.

You can globally remove the Print and/or Export buttons from the Preview window by setting the following options in the **DataLink_Viewer.ini** file:

```
[Options]
Disable_Print_Button=TRUE          (note: this disables all 3 print buttons)
Disable_Print_ThisPage_Button=TRUE
Disable_Print_Quick_Button=TRUE
Disable_Export_Button=TRUE
Disable_Search_Button=TRUE
Disable_Refresh_Button=TRUE
```

Alternatively, you can control these options via **command line arguments**. For example (all in one line):

```
"C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe" -v
"C:\temp\MyReport.rpt.rpt" "Parm1:1997" "Disable_Print_Button:TRUE"
"Disable_Export_Button:TRUE"
```

Note: disabling the Search Expert button, doesn't remove the simple search button. It removes the more advanced search button that allows users to actually browse into data fields.

Note: see “*Enforcing Settings in a Master DataLink_Viewer.ini File*” for information about enforcing these settings from the “master” ini file to all “user” ini files.

Disabling DataLink Viewer Buttons

In some scenarios you may wish to disable certain user interface buttons so that, for example, users can't check for online updates, convert rpt files to rpz files, open the user manual, etc.

You can disable user interface buttons by setting the following options in the **DataLink_Viewer.ini** file:

```
[Options]
Disable_RPZ_Creation=TRUE
Disable_Check_for_Updates=TRUE
Disable_Options_Dialog=TRUE
Disable_Browse_Dialog=TRUE
Disable_User_Manual=TRUE
Disable_Version_Info=TRUE
```

Note: see “*Enforcing Settings in a Master DataLink_Viewer.ini File*” for information about enforcing these settings from the “master” ini file to all “user” ini files.

Add your Company Info to the About Dialog

In DataLink_Viewer.ini you can **set 3 lines in the About dialog to text of your choice**. This can be useful in a large company where you want users to know **who to contact with technical questions**. It can also be useful when you sell Crystal reports bundled with DataLink Viewer and you wish to specify your contact information for similar reasons.

For example, using the following settings in DataLink_Viewer.ini:

[Options]

About_Line1=www.acme.com

About_Line2=For Technical Support, contact Jane Doe:

About_Line3=Jane_Doe@acme.com (888) 1234-4567

you would get the following About dialog:

Note: as demonstrated with the **www.acme.com** line, DataLink Viewer automatically assigns a web link to lines that contain only a url.



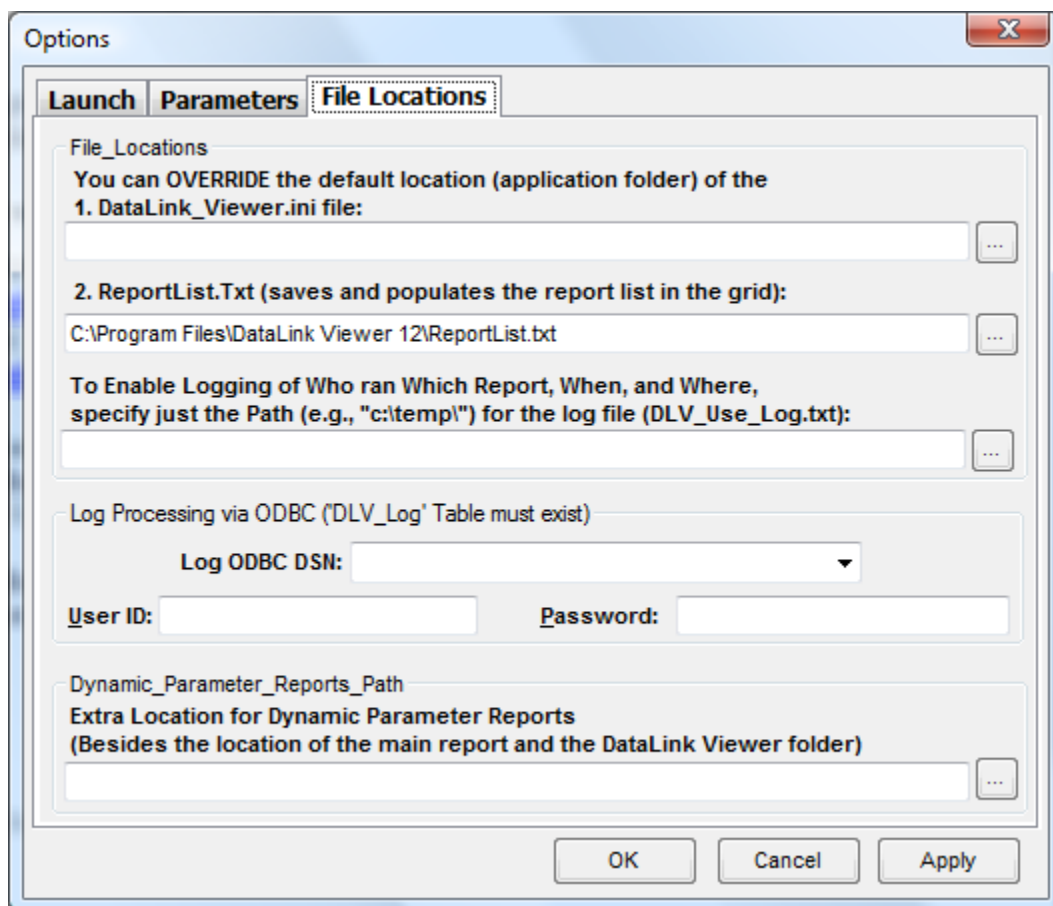
Citrix and File Location Functionality

DataLink Viewer can be installed on Citrix but be sure to install from the Console.

The **File Locations** tab in the Options dialog **allows a single installation of DataLink Viewer on a Citrix or Terminal Server to support individual settings for each user** by providing each user their own version of **DataLink Viewer.ini** and **ReportList.txt** file under their own mapped user drive. Note: upon initial launch of DataLink Viewer by a user, if the target **folders** don't exist, they get created, and if the target **files** don't exist, they get copied from the master copies in the application folder.

An alternative approach is to use these settings to force all users to share a centralized version of these files. In such a case, you should make the ReportList.txt folder read only to the users, so that the master list of reports doesn't get changed by the users.

This dialog also allows you to specify a **centralized/local logging of report use statistics**.



Explicit Assignment of Default Printer

To address a rare printing failure scenario under Citrix and Windows 2003, the following Datalink_Viewer.ini option allows an explicit assignment of the default printer when the user clicks the print button (before the printer selection dialog is displayed):

[Options]

Set_Report_To_Default_Printer=TRUE

File Location & Redirect Logic

Each time DataLink Viewer loads, if the user doesn't have write permissions on the **DataLink_Viewer.ini** file in the application folder (typical for Vista & Windows 7 machines), the location of the Master DataLink_Viewer.ini is redirected by following these steps:

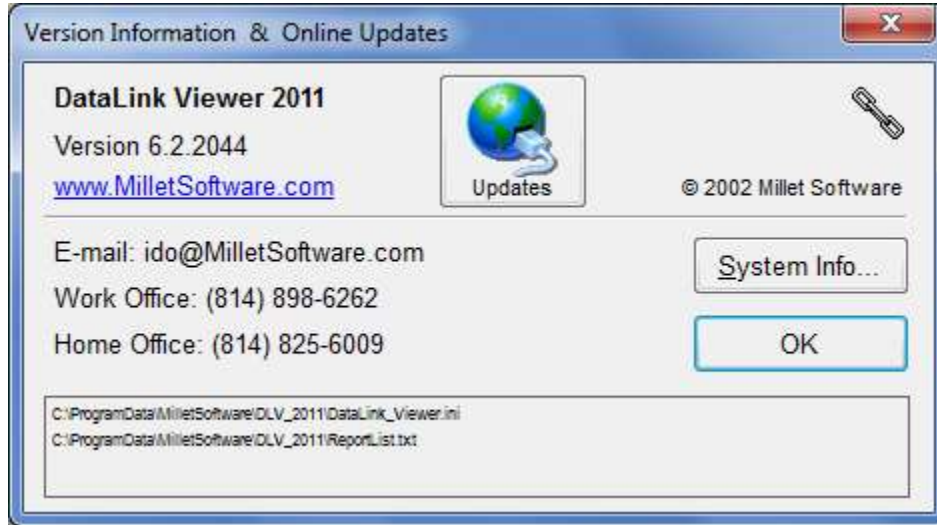
1. If the ini file in the application folder has a value specified for "ini_file" in the [File_Locations] section, that location is used for the DataLink_Viewer.ini
note: this allows an administrator who has write permissions on the application folder to manually set the location.
2. More typically, the value in step 1 would be found as blank. DataLink Viewer then determines if a redirect is needed by testing to see if: a) ReportList.grd doesn't exist in the application folder **or** b) user doesn't have write permissions to DataLink_Viewer.ini in the application folder. If a redirect is needed, the application tries to locate and use DataLink_Viewer.ini in one of the following app data folder locations, under a **\MilletSoftware\DLV_2011** folder:
 - a. **Common App Data folder** - that location allows all users on that local machine to share the same ini file
 - b. **User's Local App Data folder** - that location allows each user on that machine to have their own ini file
 - c. **User's Roaming App Data folder** - that location allows each user to have the same ini file "follow them" to other machines.
3. If DataLink_Viewer.ini was not found in any of the 3 locations above, which is **typical in 1st-time installation scenarios**, DataLink Viewer will copy it as well as ReportList.txt from the application folder to the **Common App Data folder** and notify the user that the redirect has occurred. The location of ReportList.txt would also be set to the same redirected location.

Notes: an ini file that was redirected to one of the locations above due to a write-protected application folder is considered a Master ini file. As usual, that Master ini file can be redirected further to a slave ini file by manually setting its "ini_file" location option under [File_Locations].

A blank option for ReportList.txt location in a redirected ini file is considered as an indication that the ReportList.txt file should be automatically copied to (if missing) and used at that location.

CHECKING AND NAVIGATING TO KEY FILE LOCATIONS

To check the locations of DataLink_Viewer.ini and ReportList.txt, you can click on the **'Version Information & Updates'** button. The textbox at the bottom of the dialog shows the current file locations:



To navigate to the Folder where DataLink_Viewer.ini is located, **double-click that textbox** and DataLink Viewer will open that folder location in File Explorer. That makes it easier to find and edit the ini file in cases where the folder is hidden (typical for app data folders).

Enforcing Settings in a Master DataLink_Viewer.ini File

The “master” DataLink_Viewer.ini file is located in the application folder. As described above, the [File_Locations] section in the “master” file may indicate that a different “user” DataLink_Viewer.ini file, located at a different folder, should be used. As an administrator, you may want to ensure that some centralized settings in the “master” ini file always override the settings in the “use” ini file. You can do this by setting the values to all upper case (for example, TRUE or FALSE instead of True or False). This tells DataLink Viewer that it should use these settings from the “master” file.

This behavior applies to the following options, shown as if you wanted to apply them centrally from the “master” DataLink_Viewer.ini file:

```
[Options]
Disable_RPZ_Creation=TRUE
Disable_Check_for_Updates=TRUE
Disable_Options_Dialog=TRUE
Disable_Browse_Dialog=TRUE
Disable_User_Manual=TRUE
Disable_Version_Info=TRUE

Disable_Print_Button=TRUE
Disable_Export_Button=TRUE
Disable_Search_Button=TRUE
```

Updating DataLink_Viewer.ini via a Delta File

When deploying DataLink Viewer to many users, you may want to automate the process of updating some of the DataLink_Viewer.ini settings. You can do that by placing a **DataLink_Viewer_Delta.ini** file in the application folder. Any entries found in that file will update DataLink_Viewer.ini when the application is launched.

Here is an example of a **DataLink_Viewer_Delta.ini** file:

```
[Delta_Options]
Delete_After=NEVER
//USE or NEVER or Some Date specified as yyyyMMdd
// if entry above not found, then default is USE
Update_Master_INI=False
// if entry above not found, then default is False
Update_Slave_INI=True
// if entry above not found, then default is True

[Options]
Attempt_Logon_Without_Password=False
Strip_Table_Qualifiers=True
Saved_Data_Action=Display
Associated_File_Launch_Mode=View Only
New_Window_On_DblClick=True
Parameter_Values_Remember_Max_Chars=900
Saved_Parameter_Set_Minimum_N=5

[Integrated_Authentication]
Enable_Integrated_Authentication=True
```

The [Delta_Options] section is used only to control the following aspects of the process:

- The Delete_After option controls when the Delta ini file is deleted:
 - NEVER - the file will not be deleted
 - USE - the file is deleted after being used once
 - yyyyMMdd - the file is deleted after the specified date
- Update_Master_INI controls whether the Master ini file is updated.
- Update_Slave_INI controls whether the User ini file is updated.

Update History


Version 6.2.2070: Entered Testing May 5, 2012

- ◆ This version supports the new features in Crystal 2011:
 - **Run .rptr (read only) Crystal reports.**
 - **Export Excel (Data Only) natively to .xlsx files**
note: regular (not Data Only) excel exports to .xlsx still use custom conversion from xls to xlsx. This is because Crystal 2011 provides native export to xlsx only for Data Only exports.

- ◆ **Added Data Visualization functionality.**

For detail, see '*Data Visualizer*'



- ◆ Users can now click a new toolbar icon  (or Ctrl-G) to launch a **Group Swap Expert**. This new dialog **allows changing (via drag-and-drop) the fields/formulas used to group the report or reordering the groups.** For example, instead of Grouping the report by Country, and within Country by City, you may Group the report by Product Type and within Product Type by Product.
For detail, see '*Dynamic Grouping (Group Swap Expert)*'

- ◆ The Option dialog now has a new **Export** tab with an option to control the After- Export action. You can elect to automatically open the exported file within its associated application, Open the folder in File Explorer, take no action, or Ask the user each time they export. The dialog for asking the user has a checkbox for adopting the selected choice and applying for all future exports (to avoid being asked again).



- ◆ Added **Ctrl-C** to **copy the text** and **Ctrl-Shift-C** to **copy the tooltip text** of a clicked field in report preview. Since you can use a formula to control the content of a tooltip, this allows you to copy to the clipboard much more than just the displayed value of a field. For detail, and sample use scenario, see the user manual section on "*Copying Text Content from the Report.*"

- ◆ In the 2008 version, exporting drill-downs in rpz files resulted in exporting the main report instead. In the 2011 version, this limitation is now removed.
- ◆ Fixed rendering of parameter refresh dialogs when user elects to set Display options to larger font sizes.
- ◆ Using a command line argument: “Ignore_Saved_Parameter_Values:[ALL]” you can run a report and get prompted for unspecified parameter values, skipping the saved parameter values dialog.
- ◆ Improved handling of “Printer:Default” and “Printer:Dialog” arguments:
 - a) the main window is minimized unless user interaction with login or parameter dialog is required,
 - b) if the user cancels from a parameter dialog, the process closes without a warning message about missing parameter values,
 - c) “Printer:Dialog” now closes automatically after printing.
- ◆ A configuration file is now available at
C:\Program Files\Millet Software\DataLink Viewer 2011\DataLink_Viewer_2011.exe.config
It allows users to set number of rows per parameter “page”.
- ◆ The status bar at the bottom of the window shows how many records were selected out of the total number of records retrieved.
- ◆ Fixed a login issue with ODBC_DSN command line argument.
- ◆ Fixed an issue with Oracle_Server command line argument.
- ◆ Logging to ODBC can now handle database targets with limitations on number of characters in each field. The new mode (using bind variables to pass potentially long strings capturing the parameter values and the report SQL) is enabled by setting the following entry in the [Options] section of DataLink_Viewer.ini:
Log_ODBC_Type=Oracle
- ◆ **ODBC_DSN_From_To** now support **multiple pairs** separated by “||”. This addresses use scenarios where a report uses multiple ODBC DSNs (e.g. when the main report DSN is not the same as subreports’ DSNs).
- ◆ Fixed a problem with “Printer:...” command line argument.
- ◆ Online Version Update functionality was removed due to the component Update.exe being (mistakenly) flagged by virus protection software.

- ◆ When using the Options dialog to change the default location of ReportList.txt, if the target folder already contains a ReportList.txt file, the user can now elect to load that file into the grid or (upon exiting DataLink Viewer) overwrite the content of the file with the list of reports currently loaded into the grid.
- ◆ **Document2ini** command line argument now allows calling DataLink Viewer 2011 from another application and requesting that documentation about key file (DataLink_Viewer.ini, ReportList.txt) location as well as report parameters be written to a specified ini file.
- ◆ Added **XML_Path_From_To** command line argument, allowing users to override the stored path to XML files as data source for the report.
- ◆ Fixed an issue with **Database_Path** ini file entry or command line argument when used to change the location of Excel files as a data source.
- ◆ If an Excel or Access file data source is specified in the report via relative path, and the file exists in the same folder as the report, DataLink Viewer uses the file in the same folder as the data source.
- ◆ Paths to sample reports in the report list grid are now updated to C:\Program Files (x86)\... on first-time launch of DataLink Viewer on a 64-bit machine.
- ◆ Version Info and Options dialogs now always maintain their position in front of the main DataLink Viewer window.
- ◆ **The report grid right-click menu provides options to increase/decrease font size.** The changes are saved and reused across sessions.

KNOWN ISSUES AND LIMITATION

- The preview window doesn't provide a Select Expert option. That option is not available in the new runtime components.
- Export types are only to disk. If you wish to email a report, first export it and then use your regular email client to email the resulting file. While this approach may take a few more seconds, it also gives you more control over the attachment names and email options. Note: if you need direct export to email capabilities, you can use DataLink Viewer XI R2 or Visual CUT.

Linked Dynamic Parameters (applies only to the special DataLink Viewer linked dynamic parameter reports, not to the regular Crystal Reports dynamic parameters):

- Data value in linked dynamic parameter reports is set via the hyperlink property instead of via a different formula. This is not a limitation, but users of prior versions should be aware of this change. It requires a change in linked parameter reports design if you wish to reuse old reports.
- Footer formula can't be used to control the default value in a linked dynamic parameter reports. Similarly, a footer formula can't be used to require a value for a linked dynamic parameter report.