REPORTING CENTER

OVERVIEW

The objective of this session is to review the functionality of the reporting center UI, highlight the major concepts and discuss how to maximize its potential for your business.

During this session, the following sections of the reporting center will be discussed in detail:

Reports List Window

DTools Report

- Report Definitions
- Report Categories
- Report Groups

Report Details Window

DTools Projects Window

Running Reports

- Filtering Report Data
    - When to use an “And” join versus an “Or” join.
    - Leveraging the “In” operator
    - Filtering on Location

Report Importing and Exporting
Report Designer

Dynamic Binding Details

How Items are Grouped and Sorted (Sorting Stack)

Aggregating Items in a Proposal

REPORTS LIST WINDOW

The Reports List Window (RLW) is the primary organization mechanism for reports in the Reporting Center. Reports can be organized (or re-organized) into categories that make sense for the end users needs. Reports can also be added to “Report Groups” in the RTL. Each of the concepts represented in the RTL will be defined below.

DTOOLS REPORT

The primary object in the RLW is the report. Users can select any report in the tree to run. The report can also be edited within the reporting center by selecting a report in the RLW and right clicking (Figure 1 Figure 2). This context sensitive editing places the name of the report into "Edit" mode and allows the user to quickly rename the report as they desire. Reports can also be edited via the "Edit Report" toolbar button and "Edit Report" menu (located in the "Edit" menu). Editing a report via the toolbar or menu option allows the user to change the name, description, and category of the report (Figure 1 Figure 2).

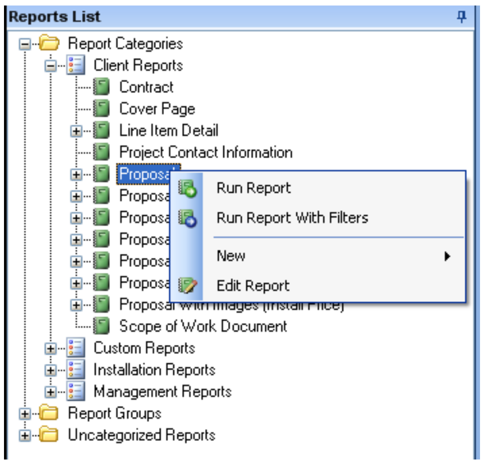
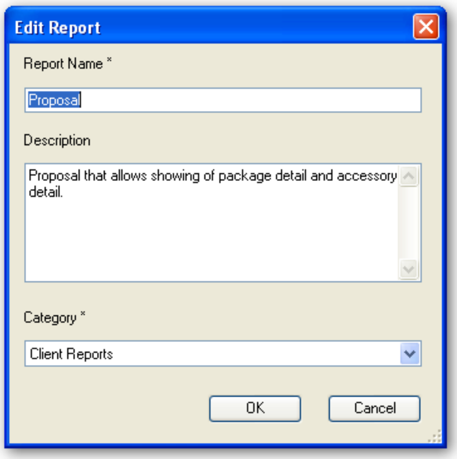


Figure 1



Report Name *

Proposal

Description

Proposal that allows showing of package detail and accessory detail.

Category *

Client Reports

Figure 2

REPORT DEFINITION

Reports that support grouping, sorting, or parameters allow for the creation, deletion, and editing of report definitions. Report Definitions represent a saved group of settings that determine how a report will generate.

The ability to create report definitions allows users to persist one or more ways they want their report to generate. The standard reports that ship with SI5 may have one or more report definitions predefined. These definitions are simply sets of configurations that D-Tools included "out of the box". These "out of the box" definitions are by no means what the user is limited to. Any user can add new definitions, edit the "out of the box" definitions, or delete them and start fresh.

ADDING A REPORT DEFINITION

Adding Report Definitions is achieved by selecting a report in the RLW and right-clicking to show the context menu and selecting New > Report Definition (Figure 3).

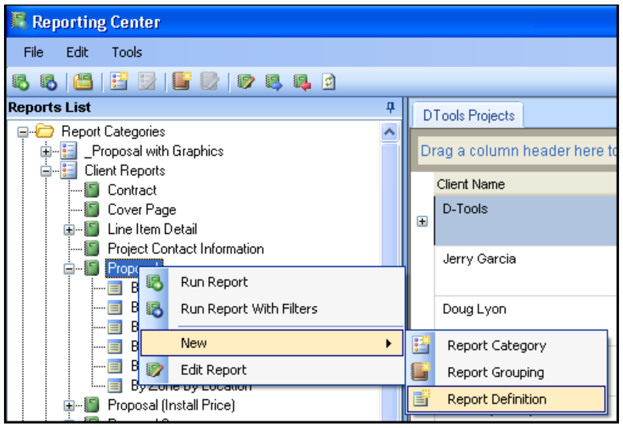
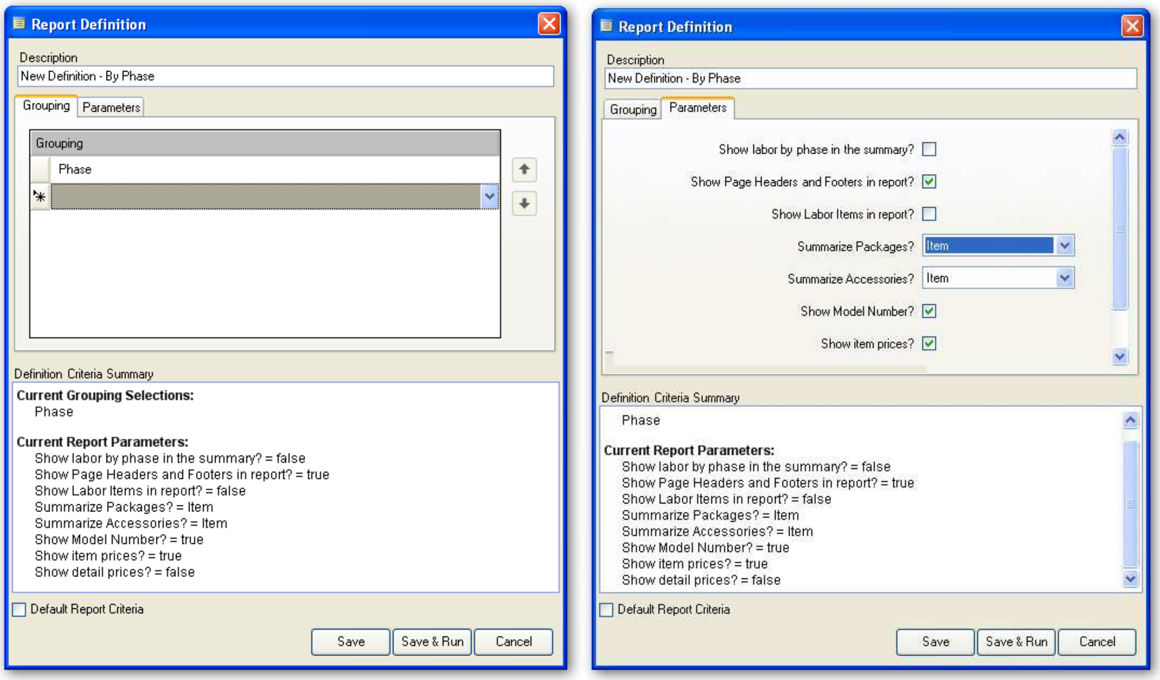


Figure 3

Each definition is required to be named. It's a good idea to give the definition a descriptive name, but bear in mind that the name of the definition does not affect the behavior of the report. The settings available will be dependent on the DTools Report (as defined in the Report Wizard). The user may have the option to set the Dynamic Group settings, Sorting options, and parameter values (Figure 4).



Description: New Definition - By Phase

Grouping: Phase

Show labor by phase in the summary? ☐
Show Page Headers and Footers in report? ✓
Show Labor Items in report? ☐
Summarize Packages? Item
Summarize Accessories? Item
Show Model Number? ✓
Show item prices? ✓

Definition Criteria Summary

Current Grouping Selections:
Phase

Current Report Parameters:
Show labor by phase in the summary? = false
Show Page Headers and Footers in report? = true
Show Labor Items in report? = false
Summarize Packages? = Item
Summarize Accessories? = Item
Show Model Number? = true
Show item prices? = true
Show detail prices? = false

☐ Default Report Criteria

Figure 4

Once the definition is saved, the user will be able to execute the report based on the settings in the definition without having to make the selections every time. This allows a user to create a definition based on how they want to see a report (show detail items or not, hide labor items, etc.).

EDITING A REPORT DEFINITION

Editing a Report Definition is achieved by selecting a Report Definition in the RLW, right clicking on the definition and selecting “Edit Report Definition” from the context menu. Once the Report Definition Dialog is open, the steps to edit an existing Report Definition are the same as adding a new one.

GENERAL NOTES ON REPORT DEFINITIONS

The concept of a “Report Definition” was created to allow users to save one or more ways a report could be generated (By Location, By Location, then Zone, etc).

Users can have add/edit/delete all report definitions.

Users can create new definitions on any report that supports one or more of the following: Grouping, Sorting, or Parameters

Any definition can be set as the “Default Definition”. This means, if the report is selected in the “Reports List” and not a specific definition, then the report will be generated based on the “default definition”.

The name of a definition has no bearing on how the report is generated and presented. If a user edits the “By Location” report definition and changes the “Groupings” to “Zone”, then the report will group the project data by zone.

Reports that support dynamic “Grouping”, most likely require a definition so the report will generate correctly with the group.

REPORT CATEGORIES

Reports in the Reporting Center may be categorized however users desire. Upon install, three categories exist for reports:

1. Client Reports
2. Management Reports
3. Installation Reports

While these categories may hit the mark for many users, others may find that they only ever run a small number of reports. In this case the user has full control to add a new category and move the reports they use to this category and delete the others.

ADDING OR EDITING REPORT CATEGORIES

Users can add report categories in one of three ways:

1. Right-Clicking on the RLW and selecting New > Report Category from the context menu.
2. Clicking the “New Report Category” toolbar in the reporting center.
3. Selecting File > New > Report Category from the main menus.

Users can edit Report Categories in one of four ways:

1. Selecting a Report Category in the RLW, right-clicking and selecting Edit Report Category from the context menu. This places the category node in “Edit Mode”.

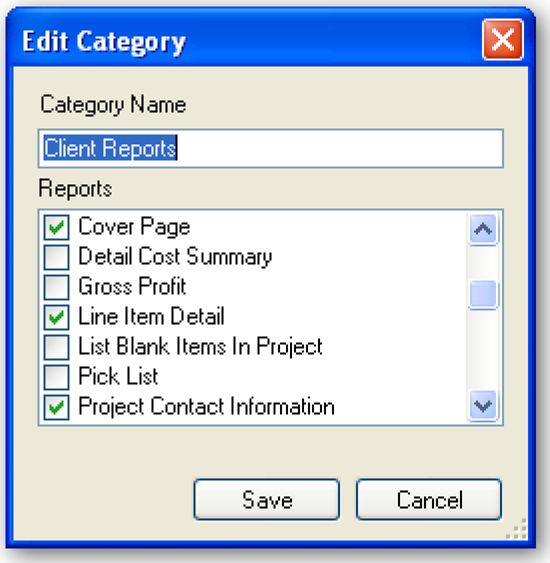
2. Single clicking on a Report Category to place it in “Edit Mode”.

3. Clicking the “New Report Category” toolbar in the reporting center.

4. Selecting File > New > Report Category from the main menus.

When adding or editing a Report Category from the context menu, the user will be presented with an editable node in the RLW to type/edit the name of the Report Category. Once the name is entered and the category is saved, users can drag and drop reports from other categories into the Report Category.

When adding or editing a Report Category from the toolbar or main menu, users will be presented with a dialog allowing them to name the Report Category and select reports to add/remove from the category (Figure 5).



Category Name: Client Reports

Reports
- [✓] Cover Page
- [ ] Detail Cost Summary
- [ ] Gross Profit
- [✓] Line Item Detail
- [ ] List Blank Items In Project
- [ ] Pick List
- [✓] Project Contact Information

Figure 5

GENERAL NOTES ON REPORT CATEGORIES

1. Report Categories are completely user definable. The Report Categories that exist “out of the box” may be edited or deleted at will based on the user needs/desires.

2. Reports can be dragged and dropped between Report Categories in the RLW.

3. When a Report Category is deleted, the reports that existed in that category will be listed in the “Uncategorized Reports” section of RLW.

REPORT GROUPS

Reports in the Reporting Center may be added to Report Groups. Report Groups function as a convenience mechanism to allow the user to run multiple reports without having to select each report separately.

ADDING OR EDITING REPORT GROUPS

Users can add Report Groups in one of three ways:

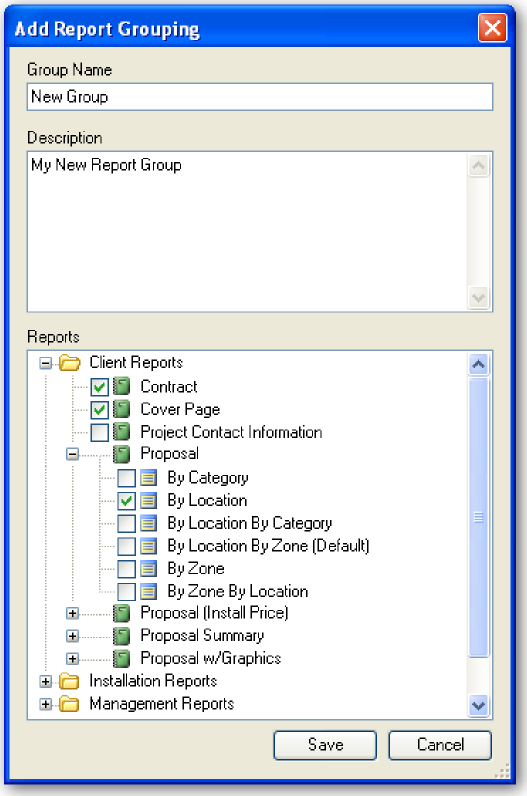
1. Right-Clicking on the RLW and selecting New > Report Group from the context menu.
2. Clicking the "New Report Group" toolbar in the reporting center.
3. Selecting File > New > Report Group from the main menus.

Users can edit Report Groups in one of four ways:

1. Selecting a Report Group in the RLW, right-clicking and selecting Edit Report Group from the context menu. This places the group node in "Edit Mode".
2. Single clicking on a Report Group to place it in "Edit Mode".
3. Clicking the "New Report Group" toolbar in the reporting center.
4. Selecting File > New > Report Group from the main menus.

When adding or editing a Report Group from the context menu, the user will be presented with an editable node in the RLW to type/edit the name of the Report Group. Once the name is entered and the group is saved, users can drag and drop reports in the RLW into the Report Group.

When adding or editing a Report Group from the toolbar or main menu, users will be presented with a dialog allowing them to name the Report Group, give it a description and select reports to add/remove from the group (Figure 6).



Add Report Grouping

Group Name: New Group

Description: My New Report Group

Reports:

- Client Reports
  - [✓] Contract
  - [✓] Cover Page
  - [ ] Project Contact Information
  - Proposal
    - [ ] By Category
    - [✓] By Location
    - [ ] By Location By Category
    - [ ] By Location By Zone (Default)
    - [ ] By Zone
    - [ ] By Zone By Location
  - Proposal (Install Price)
  - Proposal Summary
  - Proposal w/Graphics
- Installation Reports
- Management Reports

Figure 6

GENERAL NOTES ON REPORT GROUPS

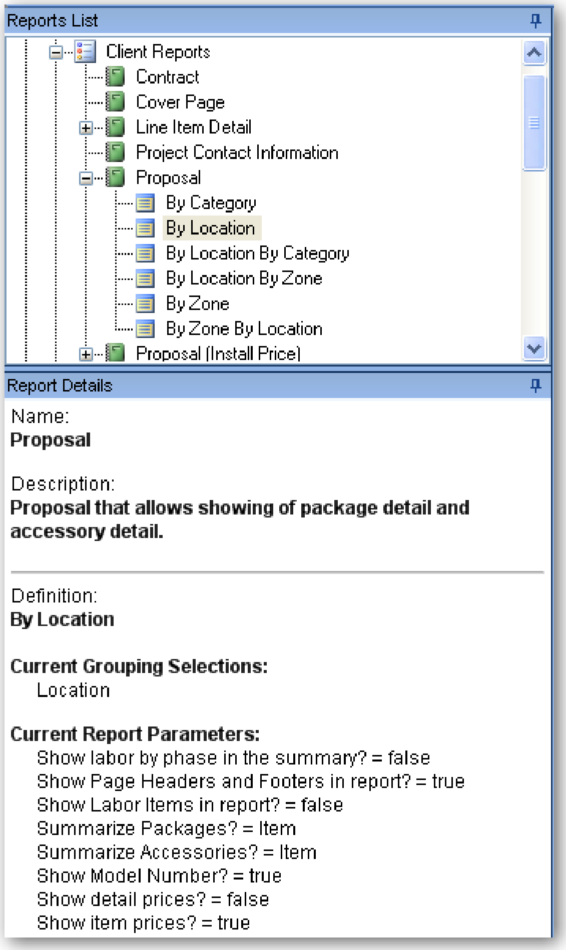
1. Report Groups are completely user definable.

2. Reports and Report Definitions can be dragged and dropped into Report Groups in the RLW.

3. When adding a report that has report definitions, it is highly advisable to add a report definition to the group and not simply the report. Reports that support dynamic grouping may not present particularly well if generated without a definition.

REPORT DETAILS WINDOW

The Report Details shows the details for a selected Report or Report Group in the Reports List Window. This window will show the name, description and settings of the selected report definition if one is selected (Figure 7).



Name:
Proposal

Description:
Proposal that allows showing of package detail and accessory detail.

Definition:
By Location

Current Grouping Selections:
Location

Current Report Parameters:
Show labor by phase in the summary? = false
Show Page Headers and Footers in report? = true
Show Labor Items in report? = false
Summarize Packages? = Item
Summarize Accessories? = Item
Show Model Number? = true
Show detail prices? = false
Show item prices? = true

Figure 7

In the Image above, one can see for the selected report (Proposal), the settings for the currently selected Report Definition (By Location) without having to select “Edit Report Definition” to view the details.

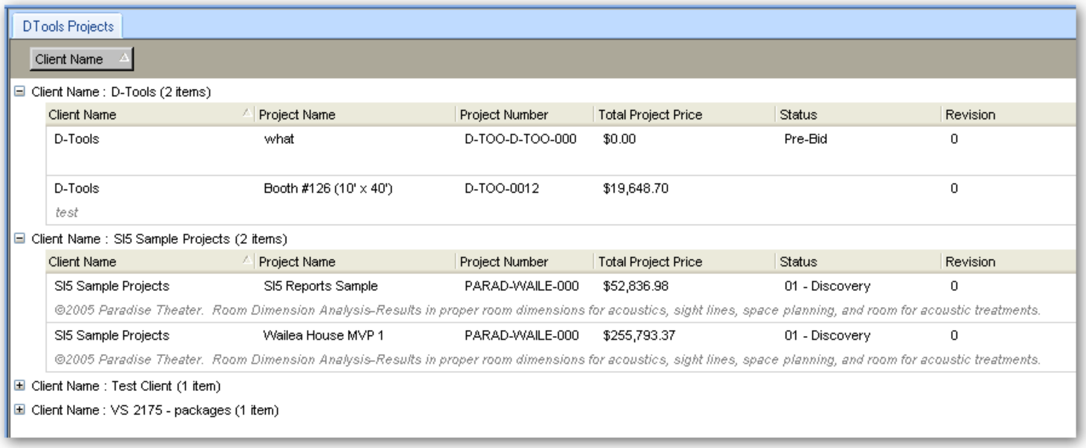
DTOOLS PROJECTS WINDOW

The DTools Projects Window (DPW) in the Reporting Center lists the projects that are available for reporting. Since the reporting in SI5 relies upon the DTL as its root data source, the only projects listed will be ones where the user has a local copy of the DTL.

Below lists a number of the features built into the DPW

The DPW allows users a quick summary view of the available projects.

Users can drag column headers to the top section to group their projects differently (Figure 8).


DTools Projects

Client Name

Client Name : D-Tools (2 items)

Client Name	Project Name	Project Number	Total Project Price	Status	Revision
D-Tools	what	D-TOO-D-TOO-000	$0.00	Pre-Bid	0
D-Tools	Booth #126 (10' x 40')	D-TOO-0012	$19,648.70		0
test					

Client Name : SI5 Sample Projects (2 items)

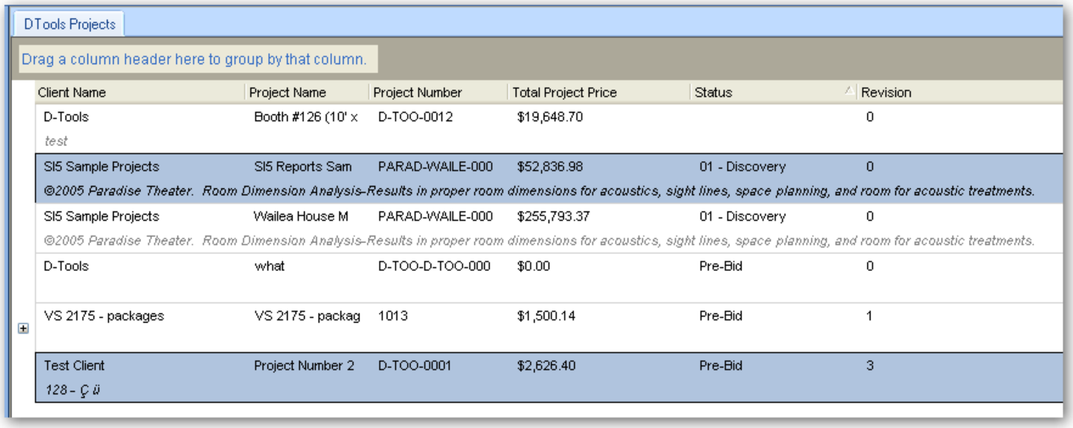
Client Name	Project Name	Project Number	Total Project Price	Status	Revision
SI5 Sample Projects	SI5 Reports Sample	PARAD-WAILE-000	$52,836.98	01 - Discovery	0
©2005 Paradise Theater. Room Dimension Analysis-Results in proper room dimensions for acoustics, sight lines, space planning, and room for acoustic treatments.					
SI5 Sample Projects	Wailea House MVP 1	PARAD-WAILE-000	$255,793.37	01 - Discovery	0
©2005 Paradise Theater. Room Dimension Analysis-Results in proper room dimensions for acoustics, sight lines, space planning, and room for acoustic treatments.					

Client Name : Test Client (1 item)

Client Name : VS 2175 - packages (1 item)

Figure 8

Users can select multiple projects to run a report against at a time (Figure 9).

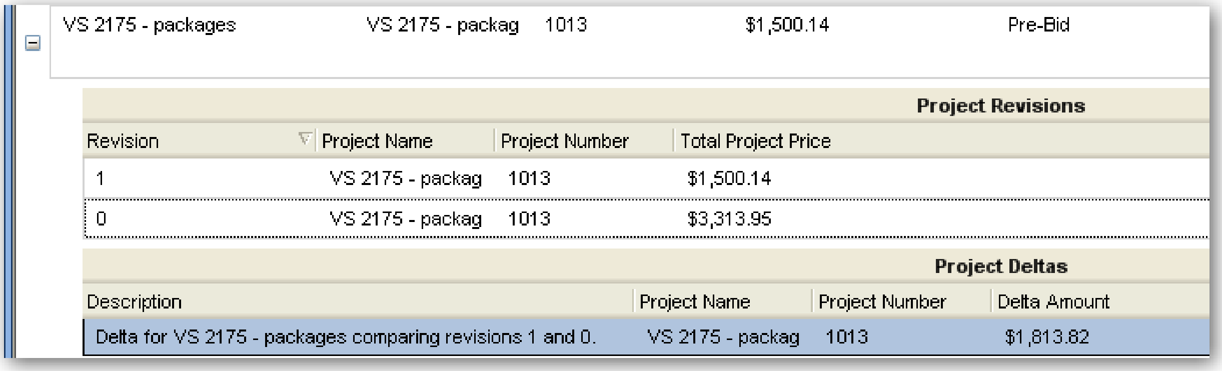

DTools Projects

Drag a column header here to group by that column.

Client Name	Project Name	Project Number	Total Project Price	Status	Revision
D-Tools	Booth #126 (10' x	D-TOO-0012	$19,648.70		0
test					
SI5 Sample Projects	SI5 Reports Sam	PARAD-WAILE-000	$52,836.98	01 - Discovery	0
©2005 Paradise Theater. Room Dimension Analysis-Results in proper room dimensions for acoustics, sight lines, space planning, and room for acoustic treatments.					
SI5 Sample Projects	Wailea House M	PARAD-WAILE-000	$255,793.37	01 - Discovery	0
©2005 Paradise Theater. Room Dimension Analysis-Results in proper room dimensions for acoustics, sight lines, space planning, and room for acoustic treatments.					
D-Tools	what	D-TOO-D-TOO-000	$0.00	Pre-Bid	0
VS 2175 - packages	VS 2175 - packag	1013	$1,500.14	Pre-Bid	1
Test Client	Project Number 2	D-TOO-0001	$2,626.40	Pre-Bid	3
128 - Ç ü					

Figure 9

- When a project has revisions (noted in the DPW by having a + next to the project row), users can create "Delta Projects" between the revisions (Figure 10).


VS 2175 - packages	VS 2175 - packag	1013	$1,500.14	Pre-Bid

Project Revisions

Revision	Project Name	Project Number	Total Project Price
1	VS 2175 - packag	1013	$1,500.14
0	VS 2175 - packag	1013	$3,313.95

Project Deltas

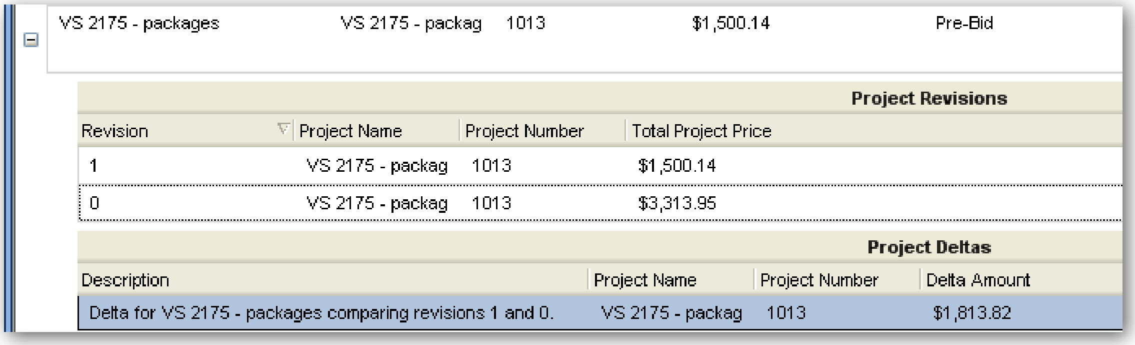
Description	Project Name	Project Number	Delta Amount
Delta for VS 2175 - packages comparing revisions 1 and 0.	VS 2175 - packag	1013	$1,813.82

Figure 10

•

To Generate a project Delta:

- Select two Revisions of a project
- Right click on the DPW.
- Click on “Create Project Delta” from the context menu.
- The Project Delta will be added to the grid (Figure 11



VS 2175 - packages	VS 2175 - packag	1013	$1,500.14	Pre-Bid

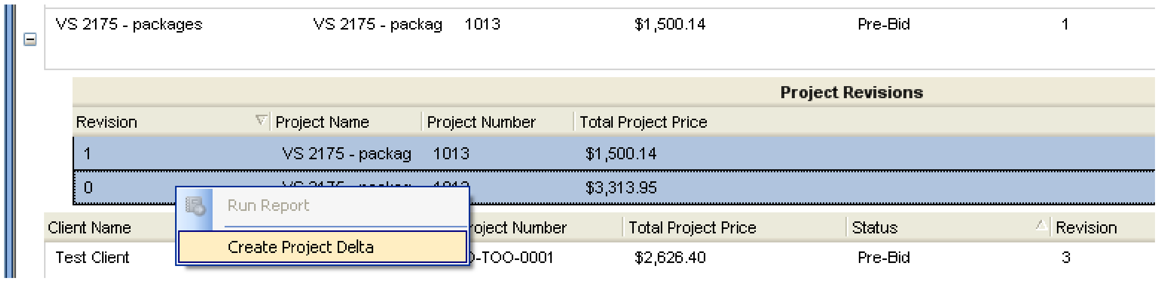
Project Revisions

Revision	Project Name	Project Number	Total Project Price
1	VS 2175 - packag	1013	$1,500.14
0	VS 2175 - packag	1013	$3,313.95

Project Deltas

Description	Project Name	Project Number	Delta Amount
Delta for VS 2175 - packages comparing revisions 1 and 0.	VS 2175 - packag	1013	$1,813.82

).



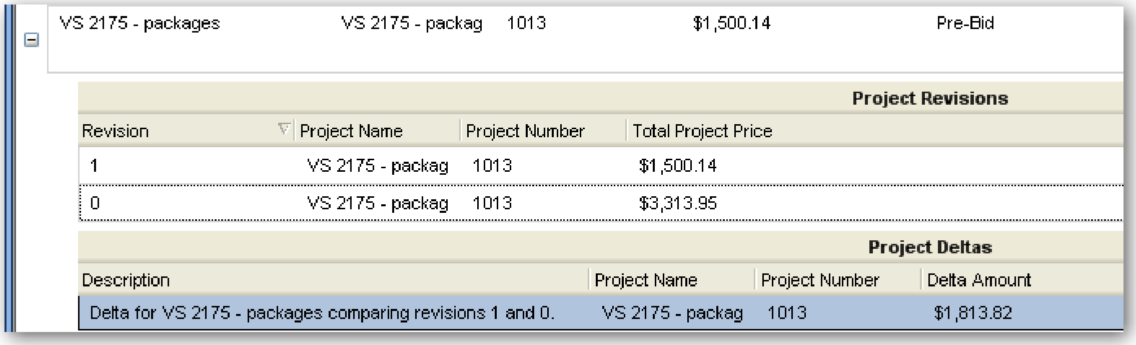
VS 2175 - packages	VS 2175 - packag	1013	$1,500.14	Pre-Bid	1

Project Revisions

Revision	Project Name	Project Number	Total Project Price
1	VS 2175 - packag	1013	$1,500.14
0			$3,313.95

Client Name		oject Number	Total Project Price	Status	Revision
Test Client		)-TOO-0001	$2,626.40	Pre-Bid	3

Figure 11



VS 2175 - packages	VS 2175 - packag	1013	$1,500.14	Pre-Bid

Project Revisions

Revision	Project Name	Project Number	Total Project Price
1	VS 2175 - packag	1013	$1,500.14
0	VS 2175 - packag	1013	$3,313.95

Project Deltas

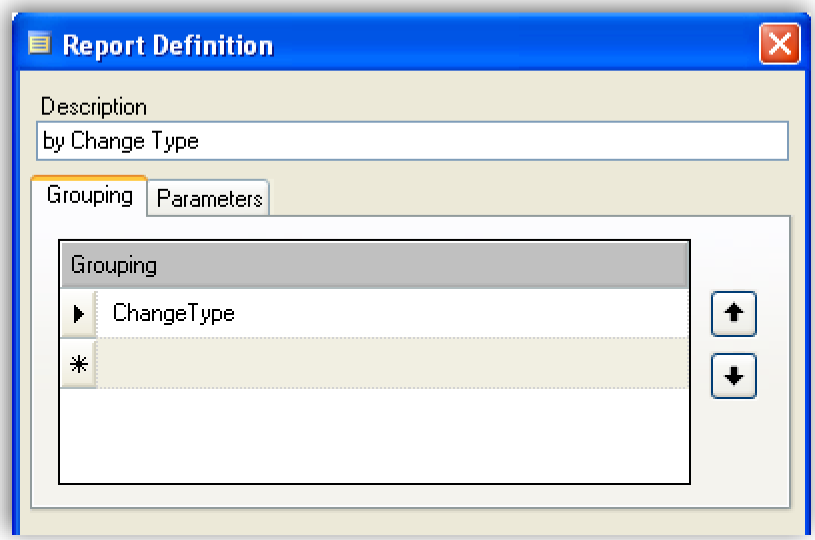
Description	Project Name	Project Number	Delta Amount
Delta for VS 2175 - packages comparing revisions 1 and 0.	VS 2175 - packag	1013	$1,813.82

Figure 12

GENERAL NOTES ON THE DTOOLS PROJECT WINDOW

Project Deltas exist for the duration that the reporting center is open.

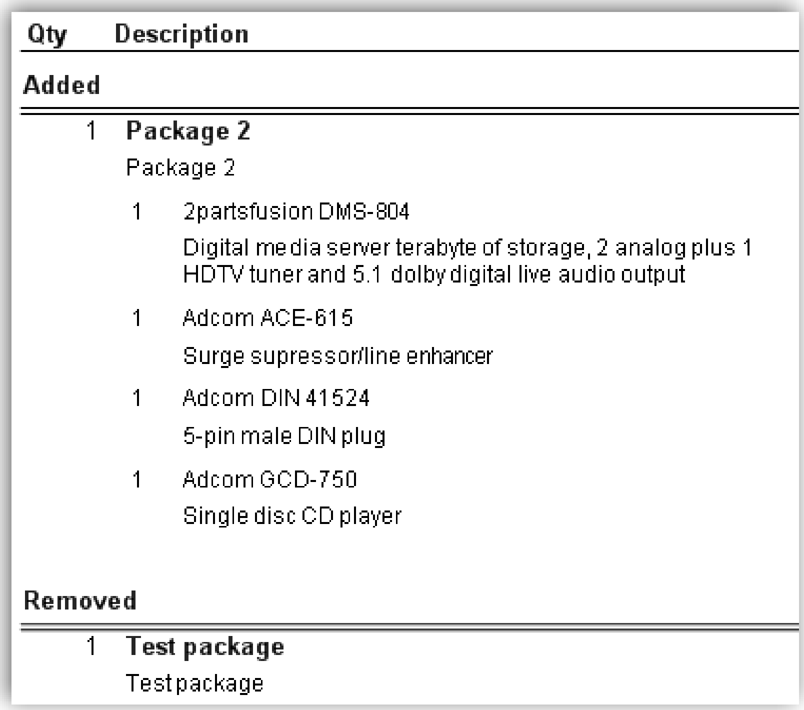
All reports in the Reporting Center may be run against the Project Deltas. Users can create a Report Definition for a Proposal that groups by "Change Type" to display items in the delta organized as Added, Removed, or Adjusted (Figure 13 & Figure 14).



Description: by Change Type

Grouping: ChangeType

Figure 13



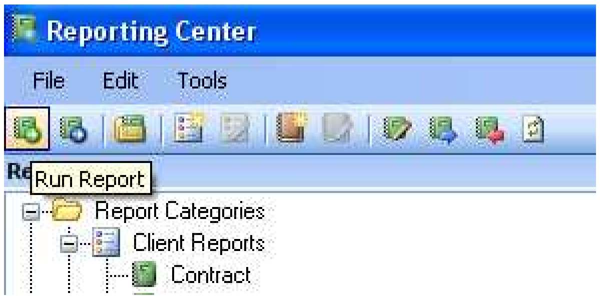
Qty	Description
Added	
1	**Package 2**
	Package 2
	1 2partsfusion DMS-804
	Digital media server terabyte of storage, 2 analog plus 1 HDTV tuner and 5.1 dolby digital live audio output
	1 Adcom ACE-615
	Surge supressor/line enhancer
	1 Adcom DIN 41524
	5-pin male DIN plug
	1 Adcom GCD-750
	Single disc CD player
Removed	
1	**Test package**
	Test package

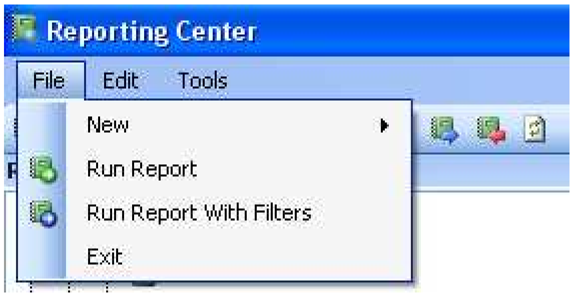
Figure 14

RUNNING REPORTS

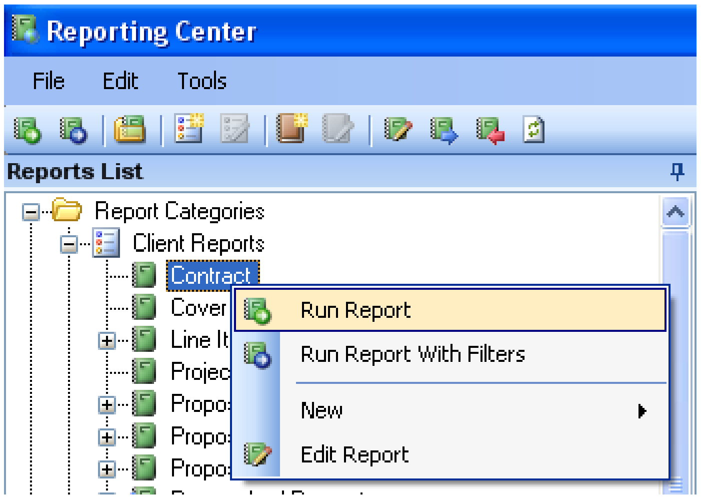
To run reports in the Reporting Center, users simply need to have at least one report and one project selected. Initiating the generation of a report can be done in the following ways:

Main Toolbar and Menus



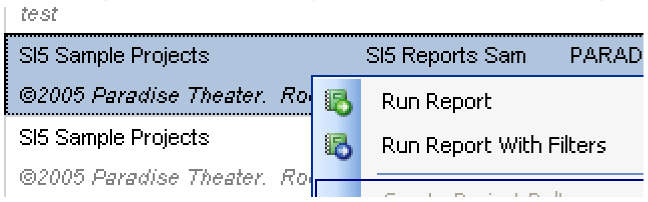


- Right-Clicking on the Reports List Window when a Report or Report Definition is selected and clicking the "Run Report" context menu.



- Double-Clicking on a Report or Report Definition in the Reports List Window.

- Right-Clicking on the DTools Projects Window and selecting "Run Report" from the context menu.



FILTERING REPORT DATA

Reports that support filtering allow users to execute the report "With Filters". This is available via all the means listed above. If a report does not support filtering, the menus and toolbars for this functionality will be disabled.

When a user chooses to filter the reports data, the user will be presented with the following dialog (Figure 15).

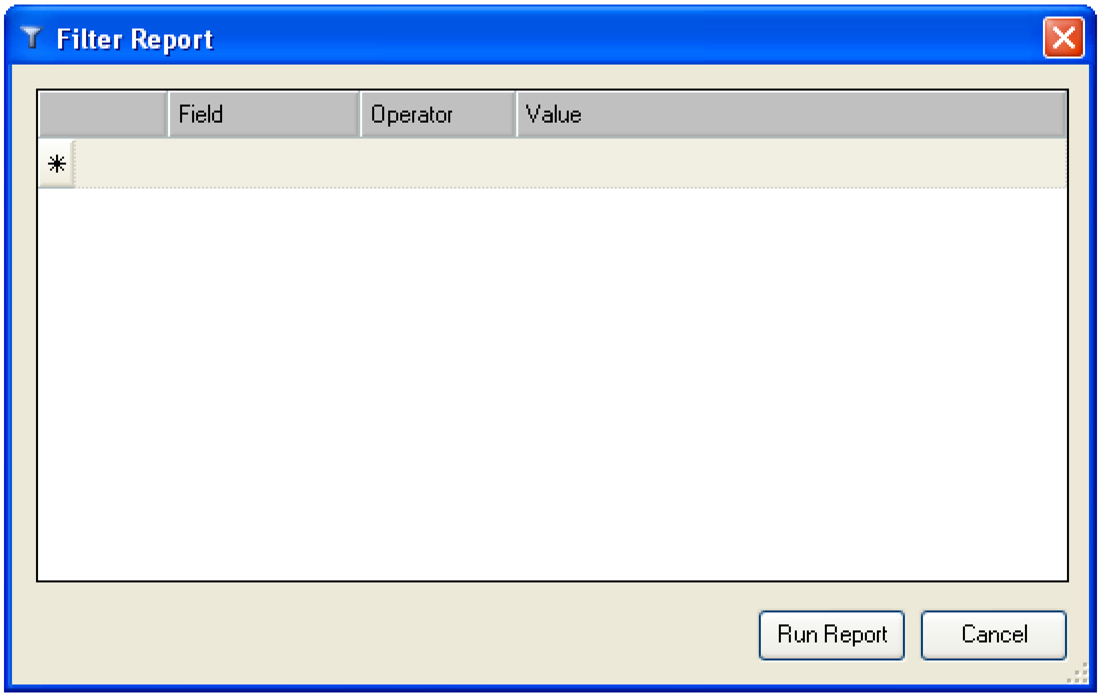


Figure 15

Users can select a field (or fields) to filter the report results. The filterable fields are properties on the items in the project. Some examples are Manufacturer, Model, Phase, Category, etc.).

Depending on the Field selected, users will have a number of "Operators" to choose from. Operators determine how the filter will behave. Whether users want to see items where a field is equal to a specific value (i.e. Manufacturer = "Sony").

Some Operators are:

Equal To

Not Equal To

Starts With

Ends With

Contains

In

Less Than (Less Than Or Equal To)

Greater Than (Greater Than or Equal To)

The list of operators will be determined by the Field selected. For instance, Fields that are textual comparisons (Manufacturer, Model, etc.) support "Starts With", "Ends With", and "Contains", but fields with numeric values (Price for instance) do not as numbers are not compared by character values like text (Figure 16 & Figure 17).

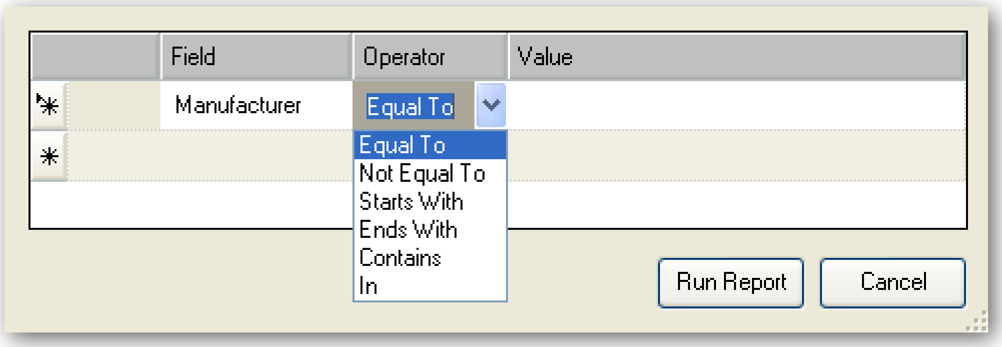


Figure 16

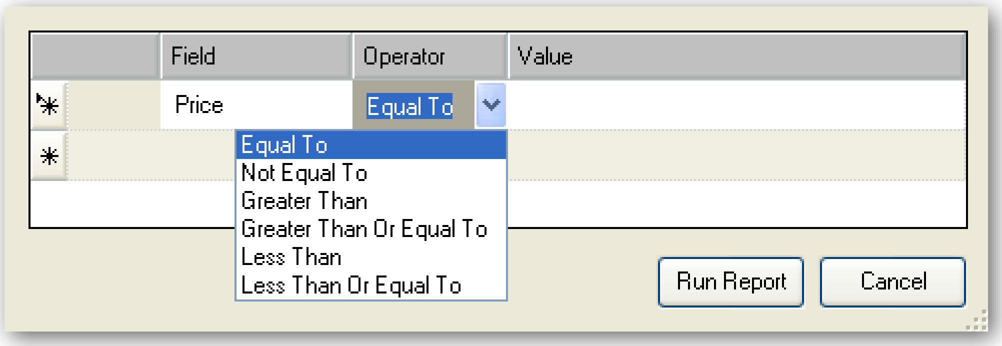
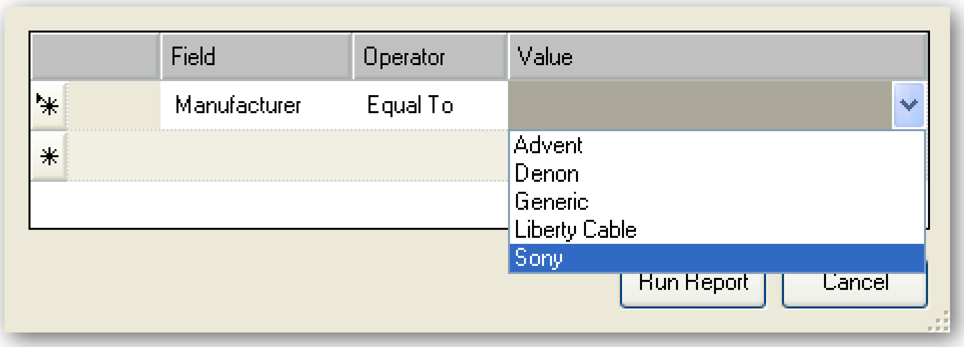


Figure 17

When possible, a list of valid values will be presented to the user to select from. For instance, when a user wants to only show all items that from a specific manufacturer and the Operator is "Equal To", if available, drop down list of all manufacturers in the project will be accessible (Figure 18).



	Field	Operator	Value
	Manufacturer	Equal To	

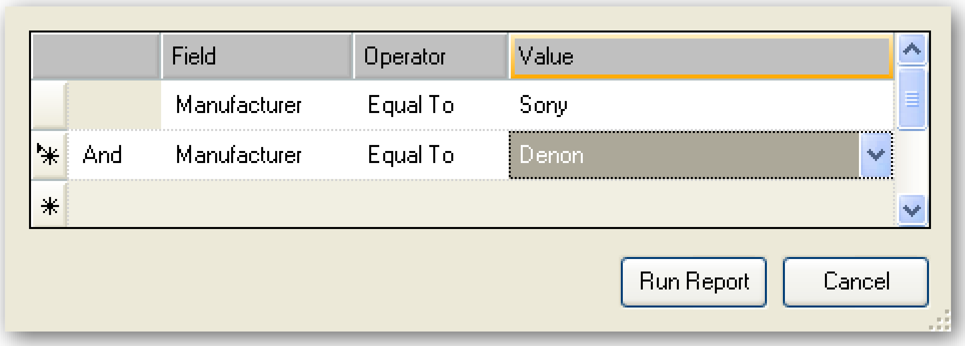
Figure 18

WHEN TO USE AN "AND" JOIN VERSUS AN "OR" JOIN.

A common confusion that has arisen when running reports with filters is the type of Join when filtering on multiple fields. People generally tend to speak in terms of "Ands". This can lead to filtering out all items in a project.

Use Case:

A user wants to run a line item report and only show items that are manufactured by Sony **and** Denon. This may lead the user to run the report with filters and define the following filters (Figure 19):

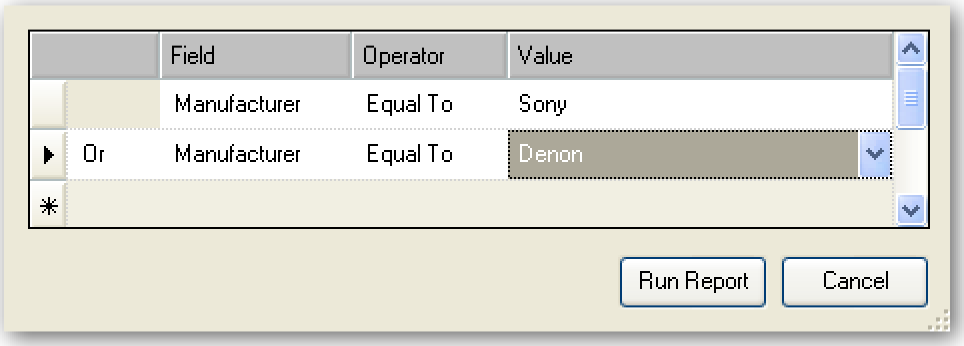


	Field	Operator	Value
	Manufacturer	Equal To	Sony
And	Manufacturer	Equal To	Denon

Figure 19

The problem with this is that both of the conditions (filters) defined are applied against each item being tested. Subsequently, an item manufactured by Sony will pass the first test, but fail the second because it is **NOT** manufactured by Denon. The above filters say: Return all items where the Manufacturer of that item is Sony and Denon. In the case of products, they only have one manufacturer so no items in the project pass the test.

From a search perspective, what needs to take place is that the items must be either manufactured by Denon OR Sony (Figure 20).



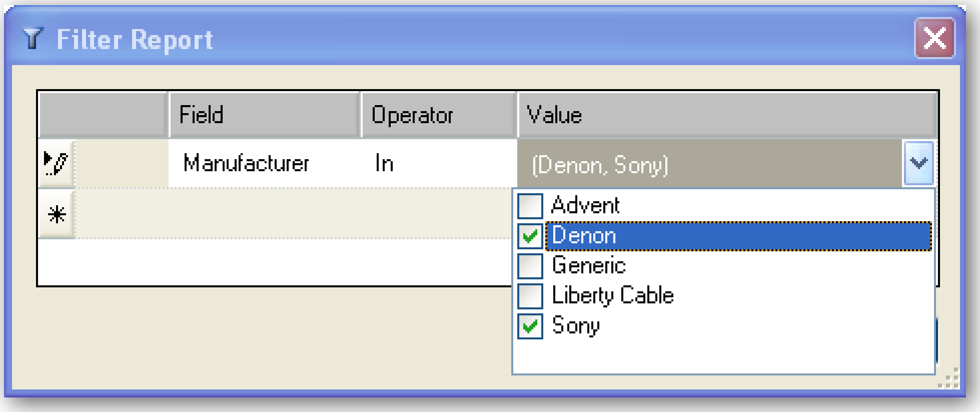
		Field	Operator	Value
		Manufacturer	Equal To	Sony
▶	Or	Manufacturer	Equal To	Denon
✳				

Figure 20

This list of filters (by changing the join to "OR") tests an item against manufacturer "Sony", if it is a Sony item, it passes the test, otherwise it tests manufacturer against Denon. If it is a Denon item, then it passes the test; otherwise, the item does not pass.

LEVERAGING THE "IN" OPERATOR

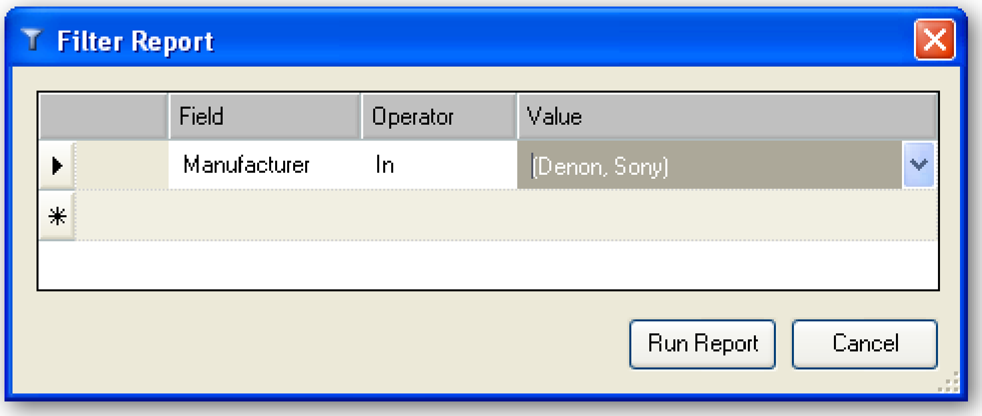
The previous section highlighted the confusion of using multiple filters with the same field. In many cases, this can be simplified by using the "In" operator. The "In" operator is a way to create a single filter for a field where it may match multiple values (Figure 21 & Figure 22). It effectively tests the value of field (in the figures below manufacturer) against a list of values. If the manufacturer of the item matches any of the selected values in the list, the item passes the test.



		Field	Operator	Value
		Manufacturer	In	(Denon, Sony)
✳				

- [ ] Advent
- [✓] Denon
- [ ] Generic
- [ ] Liberty Cable
- [✓] Sony

Figure 21



	Field	Operator	Value
▶	Manufacturer	In	(Denon, Sony)
*			

Figure 22

FILTERING ON LOCATION

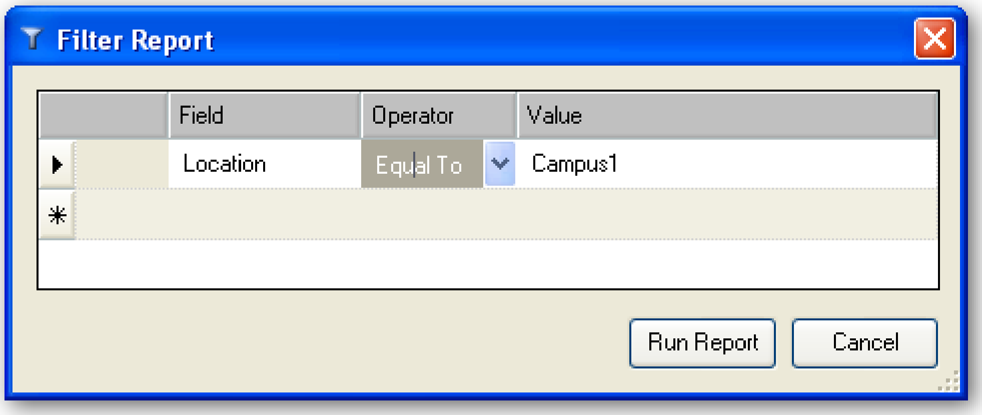
Filtering reports based on Location has some unique considerations that bear reviewing. Locations within an SI project are represented hierarchically. Each project has one or more top level locations and each of these may have child locations which in turn have child locations, etc.

Items can be associated with any location at any level. Because of this, using the "Equal To" may not return the results expected. To demonstrate this issue, consider the location/item structure below (Locations are bold, items are red):

Campus1

- Item 1
- Item 2
- **Building 1**
  - Item 3
  - Item 4
- **Building 2**
  - Item 5
  - Item 6

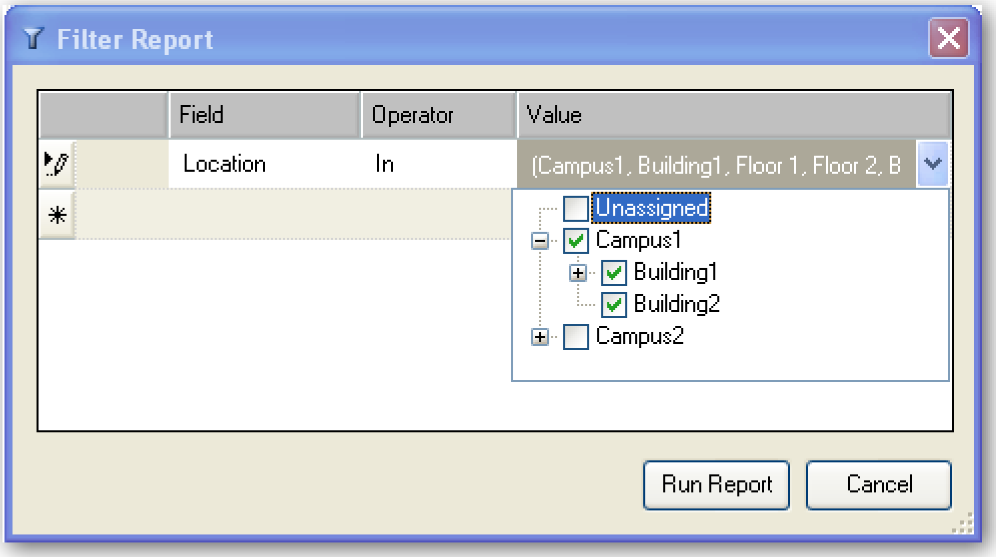
When a user creates a filter where Location = Campus1 (Figure 23), only Items 1 and 2, will be returned. This is due to the fact that while Items 3, 4, 5, and 6 exist in a sub location of Campus 1, they do not exist solely in Campus 1.



Field	Operator	Value
Location	Equal To	Campus1

Figure 23

The better approach for filtering reports based on locations is, once again, to use the "In" operator. When a user creates a filter for location in values, they are presented with a multi-select view of the locations in the project. When a parent location is selected, all child locations are selected by default (Figure 24). Users can opt out of lower level locations, but by default when a user selects "Campus1", Buildings 1 and 2 will be selected too.



Field	Operator	Value
Location	In	(Campus1, Building1, Floor 1, Floor 2, B

Figure 24

GENERAL NOTES ON FILTERS

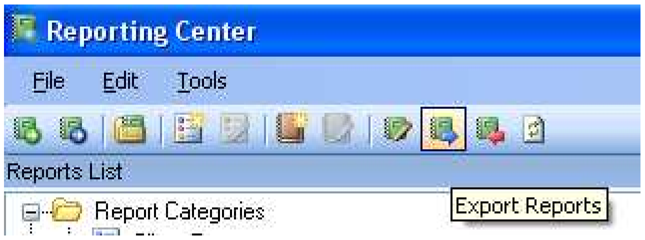
Filters are not persisted between reports. This is due to the project specific context of the filter values. Persisting filters for a report would yield unexpected results when a report was run against a project that the specific filters were not created. That said, users can create custom reports with "Required Filters" (explained later). These are filters that are applied against all instances of reports and should be limited to values that are not report specific.

Certain standard reports make use of this functionality. The wire reports are a great example of this. These reports have a required filter on the Category Type to return only items where this is equal to "Wire".

REPORT IMPORTING AND EXPORTING

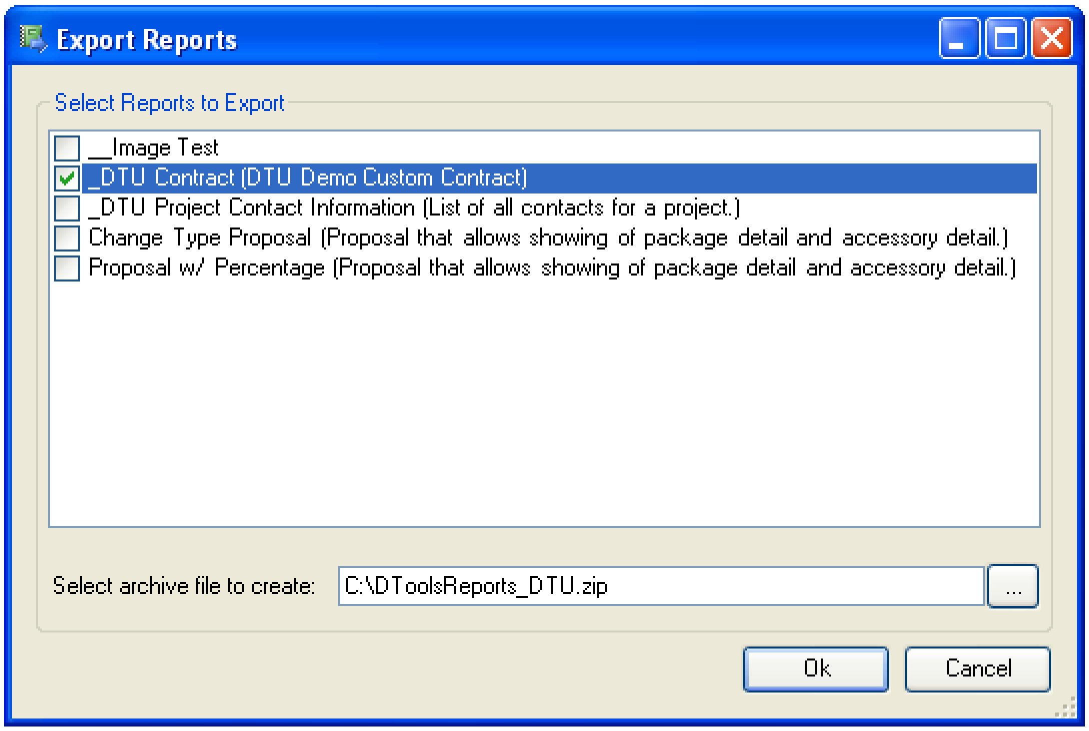
Custom Reports can be imported and exported in the Reporting Center. This provides a mechanism by which users can share and distribute reports customized to meet their business and branding needs.

Importing and Exporting is done via toolbar buttons and menus (in the Tools menu) on the main Reporting Center Window.



EXPORTING REPORTS

When a user clicks the "Export Reports" toolbar button, they will be presented with a dialog to select the reports they wish to export. Once they select the reports (one or more can be selected) and define the name and location of the archive (export) file, the main report, any custom sub reports it is dependent on and their corresponding report data files will be zipped into an archive file as defined (Figure 25).

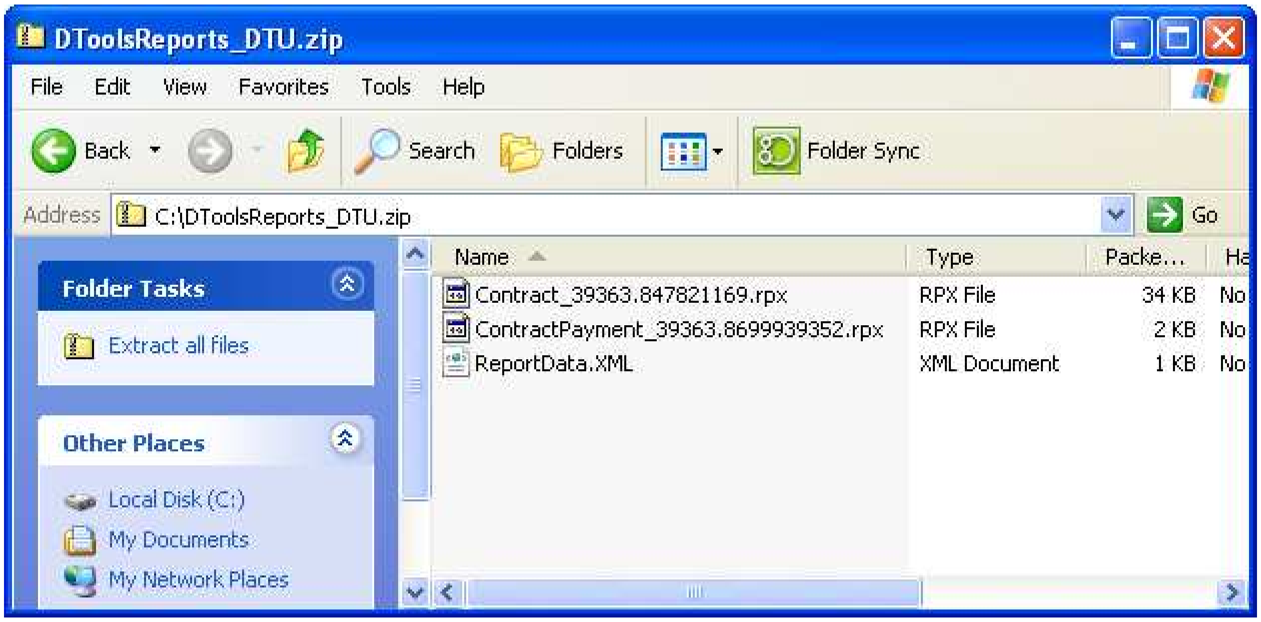


- [ ] __Image Test
- [✓] _DTU Contract (DTU Demo Custom Contract)
- [ ] _DTU Project Contact Information (List of all contacts for a project.)
- [ ] Change Type Proposal (Proposal that allows showing of package detail and accessory detail.)
- [ ] Proposal w/ Percentage (Proposal that allows showing of package detail and accessory detail.)

Select archive file to create: C:\DToolsReports_DTU.zip

Figure 25

Figure 26 shows the contents of a sample export file. Please note that the export file is nothing more than a standard zip file with specific files in it required to install this report on a different machine.



Address: C:\DToolsReports_DTU.zip

Name	Type	Packe...	Ha
Contract_39363.847821169.rpx	RPX File	34 KB	No
ContractPayment_39363.8699939352.rpx	RPX File	2 KB	No
ReportData.XML	XML Document	1 KB	No

Figure 26

IMPORTING CUSTOM REPORTS

Once a custom report has been exported, the export file can be distributed to other users via email, network share, web server, etc. Users that receive the export files can import the contained custom reports by clicking on the "Import Reports" toolbar button (Figure 27).

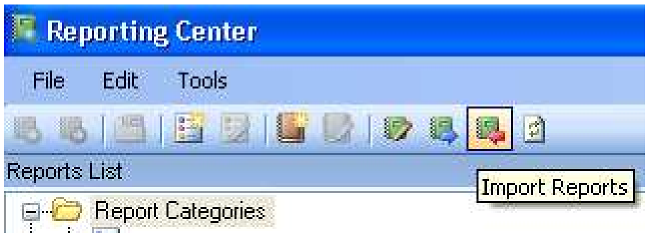
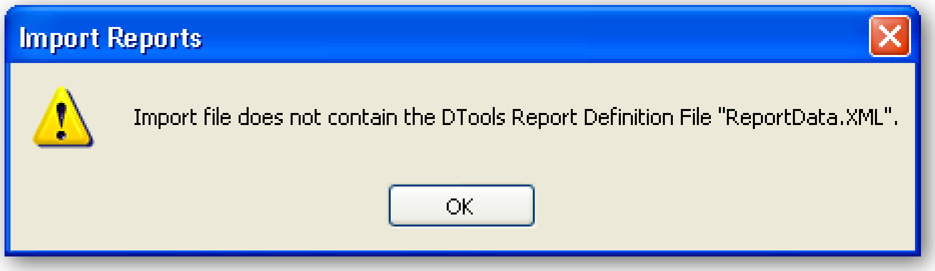


Figure 27

This toolbar opens a file browser that allows users to navigate to the Export File for import. If the user selects a ZIP file that has invalid contents, they will receive a message like the one in Figure 28.


Import Reports

Import file does not contain the DTools Report Definition File "ReportData.XML".

Figure 28

If all goes well, the report will be imported into the reports list window. Please note that there is a good change that the imported report will be located in the "Uncategorized Reports" section by default. Once imported, the report can be re-categorized as desired.

REPORT DESIGNER

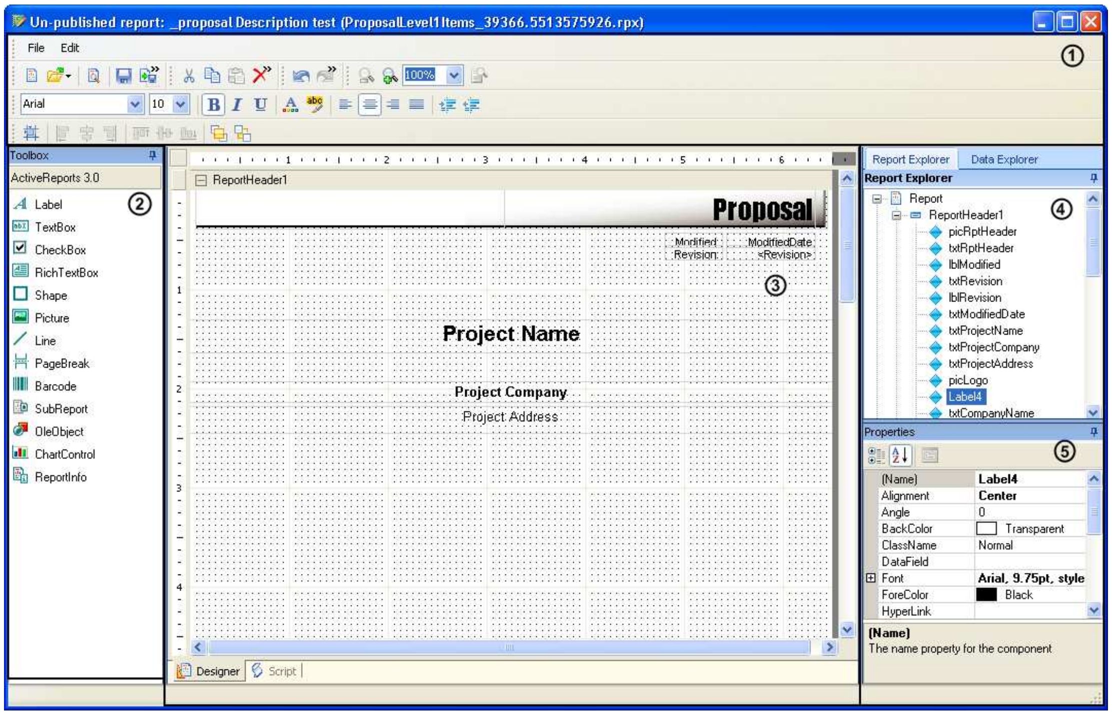
One of the most powerful, yet least documented and understood sections of the Reporting Center is the Report Designer. The Report Designer allows users to create custom reports based on existing reports or from scratch and to further modify existing custom reports.

The Report Designer UI has five primary areas (Figure 29):

1. **Menus and Toolbars** (Selecting, saving, and publishing reports and formatting components of the report loaded into the designer.
2. **Control Toolbox** – Controls available to be added to a report layout
3. **Report Designer Canvas** (Design view of the report, sections and controls. This section also provides access to the reports scripting UI.)

4. **Report Explorer and Data Explorer Window** (The Report Explorer allows users to see the sections and controls of a report in a tree format. The Data Explorer allows users to select fields from the XML structure against which the report is bound to.

5. Properties Window (Lists all the modifiable properties for the selected object in the designer, be it a control, section, or the report itself)



(Name)	Label4
Alignment	Center
Angle	0
BackColor	Transparent
ClassName	Normal
DataField	
Font	Arial, 9.75pt, style
ForeColor	Black
HyperLink	

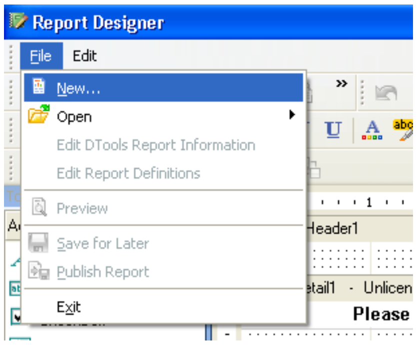
(Name)
The name property for the component

Figure 29

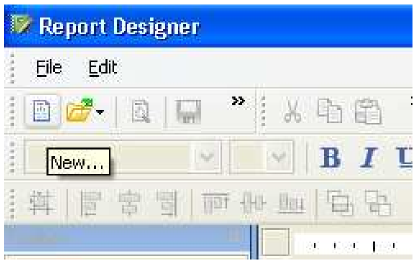
CREATING A CUSTOM REPORT

Users create custom reports by stepping through the “New Report Wizard”.

The wizard is accessible via a menu (File > New…)



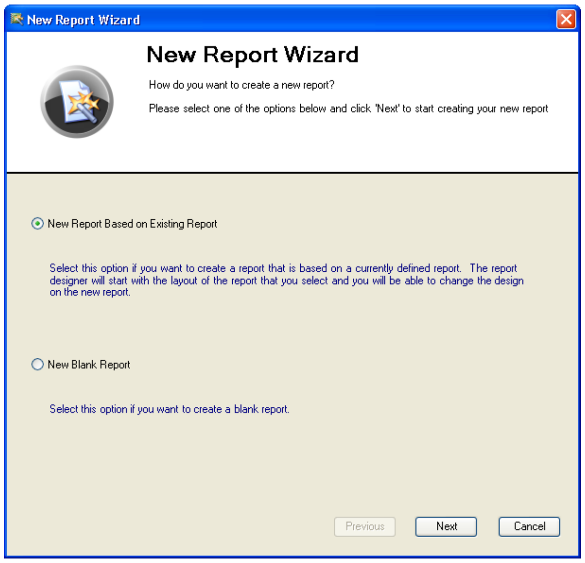
Or the Toolbar:



The Report Wizard takes the user through a number of steps; the following describes each step in Detail:

Step 1: Determine if the report will be created from the ground up or based on an existing report.

The vast majority of custom reports are created based on existing reports (the default option in this step) as most users are simply looking more closely brand the report with the other documents presented to the customer or move fields within the report.



New Report Wizard

New Report Wizard

How do you want to create a new report?

Please select one of the options below and click 'Next' to start creating your new report

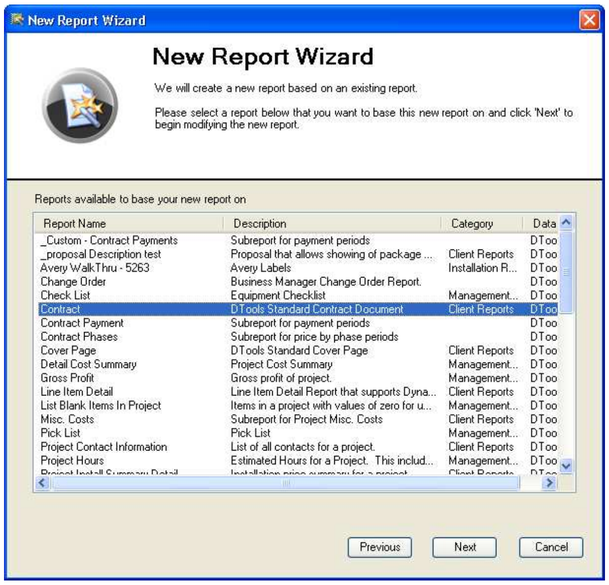
[✓ New Report Based on Existing Report]

Select this option if you want to create a report that is based on a currently defined report. The report designer will start with the layout of the report that you select and you will be able to change the design on the new report.

New Blank Report

Select this option if you want to create a blank report.

Step 1A: Should a user choose to create a report based on an existing report, the user will be asked to select a report to be used as the template. In the screenshot, the “Contract” is selected.



New Report Wizard

We will create a new report based on an existing report.

Please select a report below that you want to base this new report on and click 'Next' to begin modifying the new report.

Reports available to base your new report on

Report Name	Description	Category	Data
_Custom - Contract Payments	Subreport for payment periods		DToo
_proposal Description test	Proposal that allows showing of package ...	Client Reports	DToo
Avery WalkThru - 5263	Avery Labels	Installation R...	DToo
Change Order	Business Manager Change Order Report.		DToo
Check List	Equipment Checklist	Management...	DToo
Contract	DTools Standard Contract Document	Client Reports	DToo
Contract Payment	Subreport for payment periods		DToo
Contract Phases	Subreport for price by phase periods		DToo
Cover Page	DTools Standard Cover Page	Client Reports	DToo
Detail Cost Summary	Project Cost Summary	Management...	DToo
Gross Profit	Gross profit of project.	Management...	DToo
Line Item Detail	Line Item Detail Report that supports Dyna...	Client Reports	DToo
List Blank Items In Project	Items in a project with values of zero for u...	Management...	DToo
Misc. Costs	Subreport for Project Misc. Costs	Client Reports	DToo
Pick List	Pick List	Management...	DToo
Project Contact Information	List of all contacts for a project.	Client Reports	DToo
Project Hours	Estimated Hours for a Project. This includ...	Management...	DToo

Step 2: Begin defining the new report.

- **Name the report (Required)**. Give the report a name that makes sense to you and will make the report easily discernable from other reports. If you are creating a report based on the standard Contract, it makes sense to NOT name the report "Contract".
  Tip: When naming custom reports, add the company name to the beginning so the reports are listed together (i.e. D-Tools Contract)

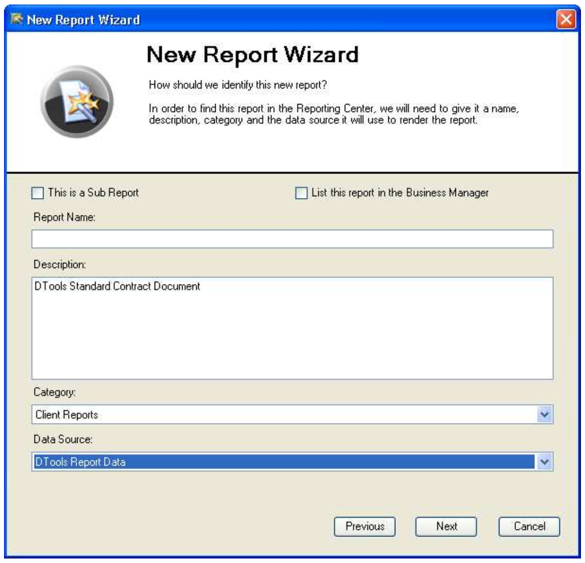
- **Set the Description (Optional)**. While this is optional it's an excellent way to provide a little more detail regarding the report. This information is displayed when the report is selected in the Report Details Window (described above).

- **Set the Category (Optional)**. Again, this is optional, but when the report is published, it will immediately be available Reports List Window in the selected category.

- **Select a Datasource (Required)**. Currently, there is only a single Datasource available, ensure that it is selected if not already.

- **Flag the Report as a "Sub Report"**. Effectively all reports can be a sub report of another report, but checking this in the wizard implies that the report is only ever executed in the context of another report. Examples of reports that have this set are the Project Summary Detail (the summary info at the end of the Proposals), the Proposal Level 2 and 3 items reports (these reports represent the child items (and grandchild items) of items in the Proposals.

- **Flag the Report as "List in Business Manager"**. When this flag is checked, the report being created will only be visible in the Change Order UI, not the main Reporting Center.


New Report Wizard

How should we identify this new report?

In order to find this report in the Reporting Center, we will need to give it a name, description, category and the data source it will use to render the report.

This is a Sub Report: [ ]

List this report in the Business Manager: [ ]

Report Name:

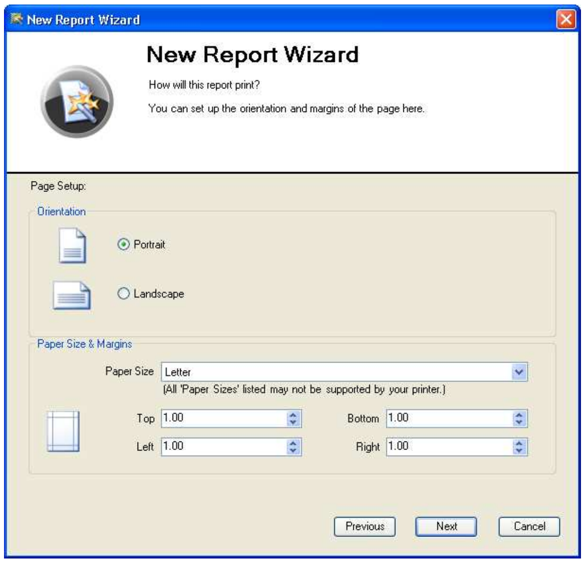
Description: DTools Standard Contract Document

Category: Client Reports

Data Source: DTools Report Data

Step 3: Set the page settings of the report.

This step allows users to define how the report is to be oriented (Portrait or Landscape), select the Paper size (Letter by Default), and set the margins (1 inch by default).


New Report Wizard

How will this report print?

You can set up the orientation and margins of the page here.

Page Setup:

Orientation: [✓ Portrait] Landscape

Paper Size & Margins

Paper Size: Letter

(All 'Paper Sizes' listed may not be supported by your printer.)

Top: 1.00

Bottom: 1.00

Left: 1.00

Right: 1.00

It is important to note that changing these settings **DO NOT** have a direct effect on the layout of the report in the designer. Changing the orientation or the margins will not automatically affect the "Width" of the report. By default, all standard reports (Letter paper size) that have the orientation set to "Portrait" have margins of one inch and a report canvas width of 6.5 inches. Changing the left and right margins to ½ inch each will not automatically change the width of the report canvas to 7.5 inches. The reasoning behind this is that reports can be designed to print in "columns". This means that one can create a layout that is (for example) 3.25 inches wide and set its Detail section can have its "ColumnCount" property set to 2. Automatically attempting to modify the width of will have adverse effects on this type of report. Furthermore, increasing the margins and automatically lowering the report

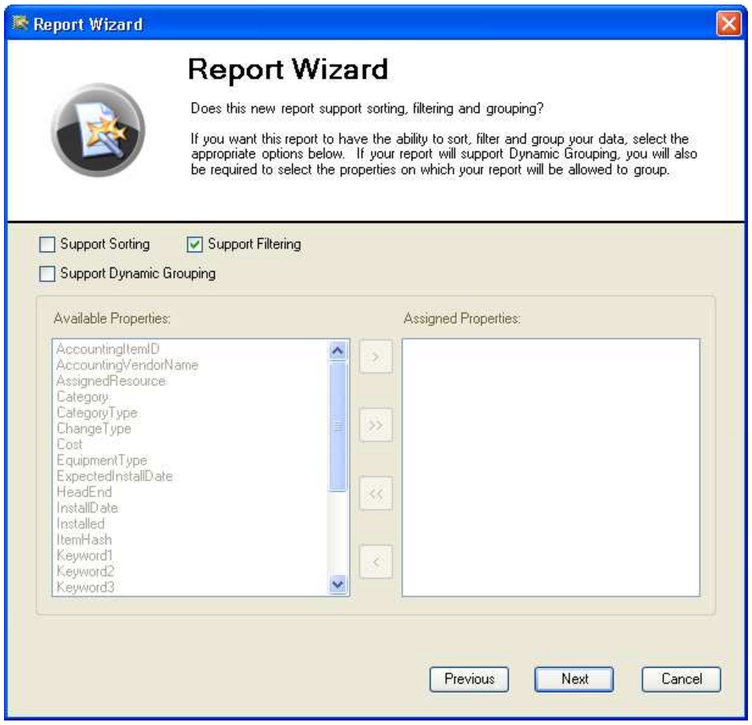
Width may push controls off the printable surface cause extra pages to print and users unable to easily determine why.

Step 4: Determine if a report should support Grouping, Sorting, or Filtering.

These settings determine control how report definitions can be created. Checking either the "Support Sorting" or "Support Dynamic Grouping" controls whether the corresponding tabs appear when creating or editing a Report Definition. Checking "Support Filtering" determines whether the "Run Report with Filters" menus will be enabled when the report is selected.

Reports that support Dynamic Grouping have specifically named sections with specifically named controls the reporting engine knows how to bind to. Also, when a report supports dynamic grouping, one can define which properties are to be available to the user when creating a Report Definition for the new report. Use the lists and toggle buttons in the wizard to add or remove available properties.

Note: Reports that do not iterate over the items should not support Grouping Sorting, or Filtering. These settings are tightly tied to items and only work when the items are the main data source of the report or a key sub report in the main report. The vast majority of reports within the Reporting Center do iterate over one of the Items sections in the XML.


Report Wizard

Report Wizard

Does this new report support sorting, filtering and grouping?

If you want this report to have the ability to sort, filter and group your data, select the appropriate options below. If your report will support Dynamic Grouping, you will also be required to select the properties on which your report will be allowed to group.

☐ Support Sorting ☑ Support Filtering
☐ Support Dynamic Grouping

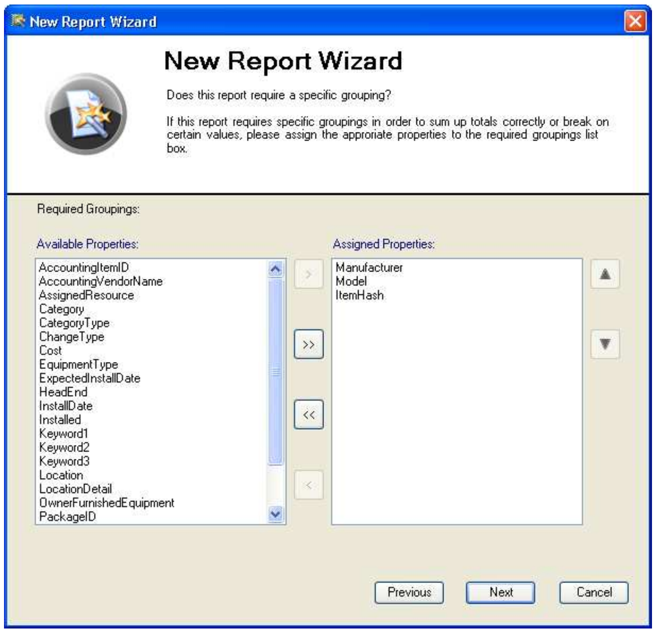
Available Properties:
AccountingItemID
AccountingVendorName
AssignedResource
Category
CategoryType
ChangeType
Cost
EquipmentType
ExpectedInstallDate
HeadEnd
InstallDate
Installed
ItemHash
Keyword1
Keyword2
Keyword3

Assigned Properties:

Step 5: Required Groupings

Some reports require that data be grouped in a specific manner in order to generate the desired results. Setting the required groupings accommodates this necessity. In the case of the Proposal Reports, regardless of how the items are dynamically grouped at run time (the Report Definition), in order for the report to aggregate properly, the items must be grouped (ascending sort) by Manufacturer, Model and

ItemHash. Grouping by Manufacturer and Model gets the like items in order together; the ItemHash property differs on like items when the child items (items in a package or accessories of an equipment item) of the like items are different.



Does this report require a specific grouping?

If this report requires specific groupings in order to sum up totals correctly or break on certain values, please assign the approriate properties to the required groupings list box.

Required Groupings:

Available Properties:

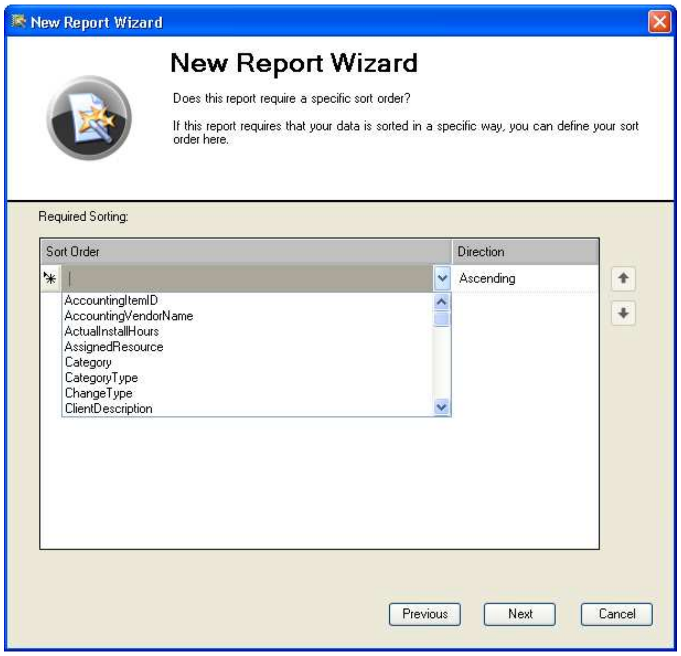
AccountingItemID
AccountingVendorName
AssignedResource
Category
CategoryType
ChangeType
Cost
EquipmentType
ExpectedInstallDate
HeadEnd
InstallDate
Installed
Keyword1
Keyword2
Keyword3
Location
LocationDetail
OwnerFurnishedEquipment
PackageID

Assigned Properties:

Manufacturer
Model
ItemHash

Step 6: Required Sorts

If the items in a report must always be ordered in a specific way, the user can define a Required Sort. Required Sorts can have minimal effects on the ordering of items depending on a number of things. For example, in a Proposal, the section that represents the details of proposal is actually a group header section that is tied to the ItemHash data field. This allows the report engine to aggregate items (sum the quantity of each item and create a single entry in the report). Since no actual detail records are being shown, a required Sort will have no effect on the presentation of the data.



New Report Wizard

Does this report require a specific sort order?

If this report requires that your data is sorted in a specific way, you can define your sort order here.

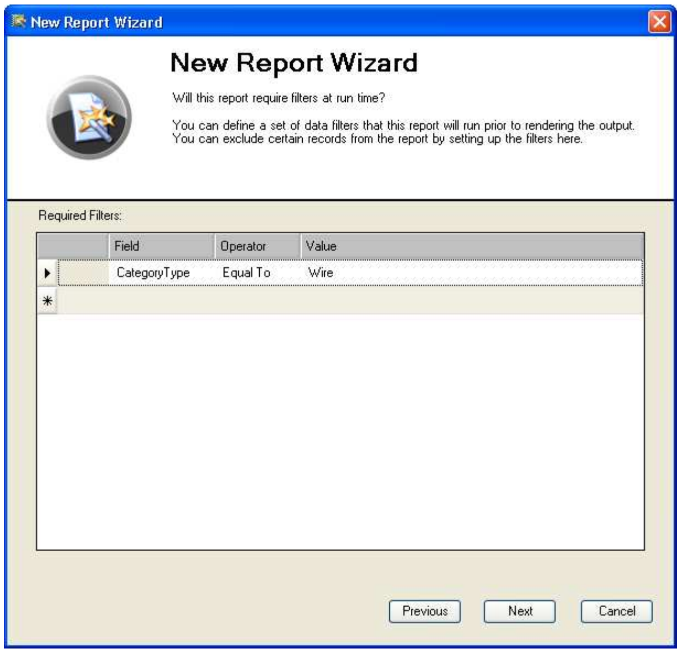
Required Sorting:

Sort Order	Direction
	Ascending

AccountingItemID
AccountingVendorName
ActualInstallHours
AssignedResource
Category
CategoryType
ChangeType
ClientDescription

STEP 7: **REQUIRED FILTERS**

Users may wish to create reports that only return data that meet specific conditions without forcing the user to run the report "with filters". This is accomplished by defining a required filter for the report. The Wire reports in the Reporting Center are excellent examples of reports that leverage this. The Wire reports only report against items where the Category Type of the item = "Wire". By defining this condition as a required Filter, all other items are automatically filtered from the report.



New Report Wizard

Will this report require filters at run time?

You can define a set of data filters that this report will run prior to rendering the output. You can exclude certain records from the report by setting up the filters here.

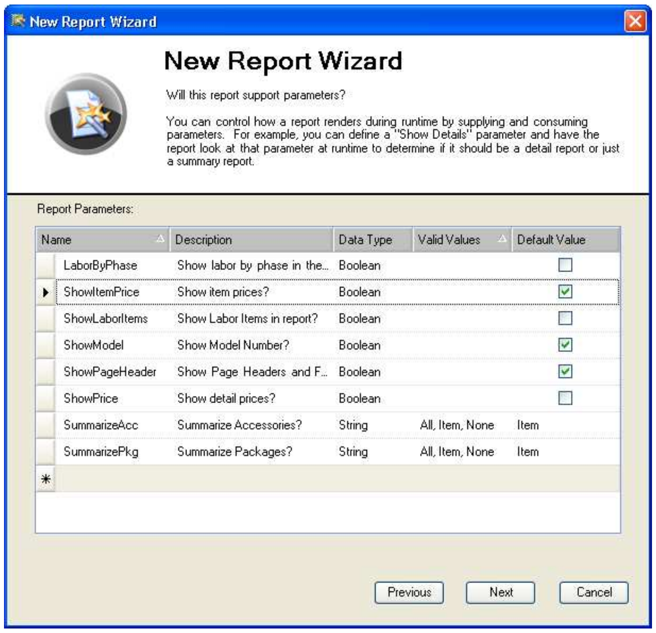
Required Filters:

Field	Operator	Value
CategoryType	Equal To	Wire

Step 8: Defining Report Parameters

Report Parameters allow for more complex control over the reports behavior. Parameters have the following parts:

- **Name** - a short string used as the key)
- **Description** – User prompt displayed in the Report Definition UI.
- **DataType** – The value type of the parameter (String, Double, Integer, Boolean)
- **Valid Values** – Values that the user can select from in the UI. This allows for better control over what can be entered in the parameters value
- **Default Value** – The value selected by default in the UI if not previously set.



New Report Wizard

Will this report support parameters?

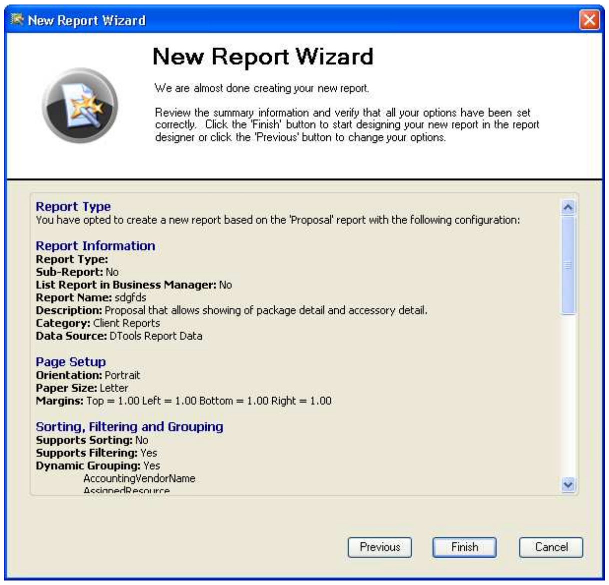
You can control how a report renders during runtime by supplying and consuming parameters. For example, you can define a "Show Details" parameter and have the report look at that parameter at runtime to determine if it should be a detail report or just a summary report.

Report Parameters:

Name	Description	Data Type	Valid Values	Default Value
LaborByPhase	Show labor by phase in the...	Boolean		☐
ShowItemPrice	Show item prices?	Boolean		☑
ShowLaborItems	Show Labor Items in report?	Boolean		☐
ShowModel	Show Model Number?	Boolean		☑
ShowPageHeader	Show Page Headers and F...	Boolean		☑
ShowPrice	Show detail prices?	Boolean		☐
SummarizeAcc	Summarize Accessories?	String	All, Item, None	Item
SummarizePkg	Summarize Packages?	String	All, Item, None	Item

While parameters are defined at this point, simply defining a parameter does not mean that it has any effect on the behavior of report. When the report is run, the value of parameter must be returned in script and used for the conditional behavior. This will be discussed later.

Step 9: Review the reports settings and save.



New Report Wizard

We are almost done creating your new report.

Review the summary information and verify that all your options have been set correctly. Click the 'Finish' button to start designing your new report in the report designer or click the 'Previous' button to change your options.

Report Type
You have opted to create a new report based on the 'Proposal' report with the following configuration:

Report Information
Report Type:
Sub-Report: No
List Report in Business Manager: No
Report Name: sdgfds
Description: Proposal that allows showing of package detail and accessory detail.
Category: Client Reports
Data Source: DTools Report Data

Page Setup
Orientation: Portrait
Paper Size: Letter
Margins: Top = 1.00 Left = 1.00 Bottom = 1.00 Right = 1.00

Sorting, Filtering and Grouping
Supports Sorting: No
Supports Filtering: Yes
Dynamic Grouping: Yes
AccountingVendorName
AssignedResource

Once the Report Layout is loaded into the designer, regardless of whether the report is based on existing a report or a completely new report, the user will be able to modify its layout.

KEY COMPONENTS IN A REPORT

- **Report** – All components of a report are children of the main report. Beyond the child components, the report itself has a number of properties that may be set. One key property described above (in the section discussing page size and margins) is the “PrintWidth” of the report. This property represents the width of the designer canvas of a report.

- **Sections** – The key section of any report built using the Active Reports engine is the "Detail" section. This section represents a single record in the data the report is iterating over. This section should be considered the middle of the report. It is the only section that does not have a sibling section. This means that the detail section of a report stands alone. All other sections (Group, Page, or report sections) have both a header and a matching footer section (Figure 30

  Sections in a report:). The sections are color coded to show the sibling sections. Notice that the Detail section (Detail1) does not have a sibling section.

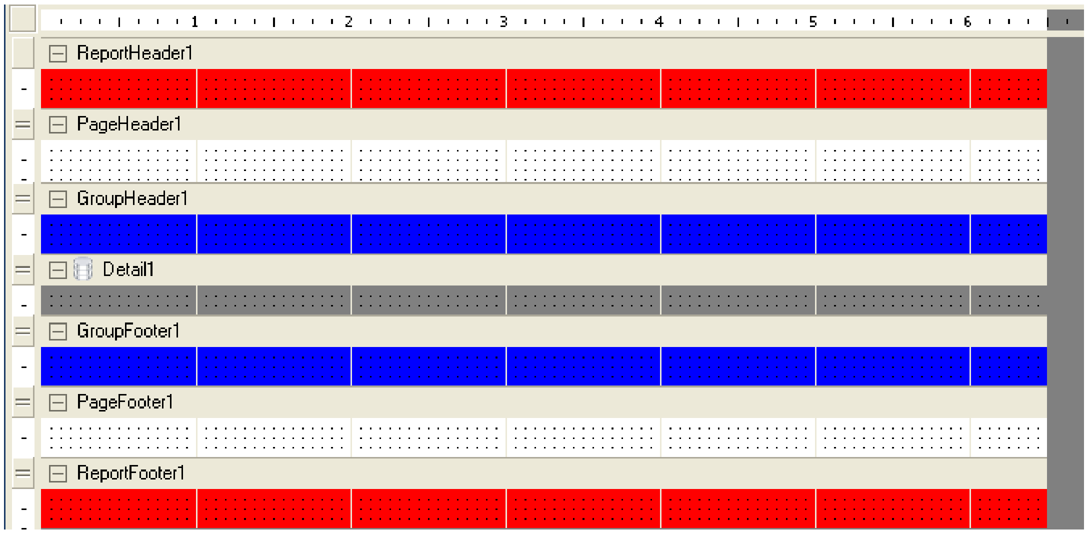


Figure 30

Sections in a report:

  - **Detail** – This is the only section that is required for a report. As stated above, this section represents a single record in the reports data.

  - **Report Header and Footer** – The report header and footer sections will print once per report. The header will be the first section rendered for the report and the footer will be the last section.

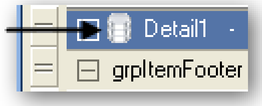
  - **Page Header and Footer** – The page header and footer sections will print once per page. The page header will always be the very first section rendered to a page in the report and the page footer will be the very last section rendered at the bottom of the page in a report.

  - **Group Header and Footer** - The Group header and footer sections represent the most powerful sections in the report. Group sections allow users to aggregate data into subtotals, counts, etc. The number of instances of a given group section in a report is determined by the number of times the data that group is bound to changes.

    Every Group Header in a report has a property called a "DataField". This property is set to the field in the data set the report is iterating over. Every time the value changes in this field, a new instance of the group is created.

In the case of SI, many times users will want to group their data in the report by location. This requires that the data in the report is iterating over be sorted by location. When the data is sorted by location, a new group will be created every time the location for the current record in the data set is different than the previous record in the data set.

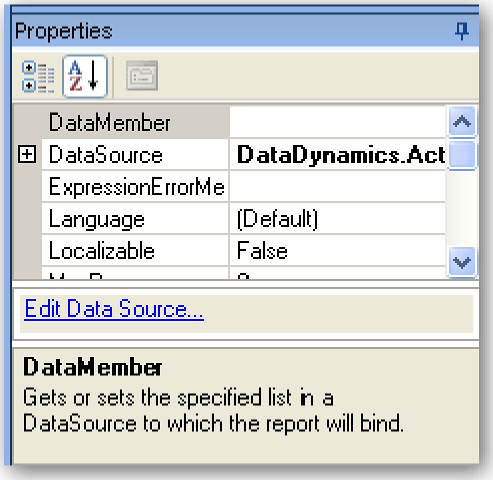
- **Report Datasource** – Every report has a defined Datasource. In the case of the SI5 reports, the Datasource is an XML document. The Datasource for a report can be modified by selecting the report in the "Report Explorer" and then clicking on the "Edit Data Source" link in the properties window (Figure 31



) or by clicking on the Datasource icon located on the Detail section of the report design canvas (Figure 32

If a user is creating a custom report based on an existing report, there is no need to modify the Datasource. By default, new custom reports NOT based on existing reports are set to iterate over the "Items" represented by the recordset pattern (XPath Expression): //dtr:Items/dtr:Item.).

If a user is creating a custom report based on an existing report, there is no need to modify the Datasource. By default, new custom reports NOT based on existing reports are set to iterate over the "Items" represented by the recordset pattern (XPath Expression): //dtr:Items/dtr:Item.



DataMember	
⊞ DataSource	**DataDynamics.Act**
ExpressionErrorMe	
Language	(Default)
Localizable	False

Edit Data Source...

DataMember
Gets or sets the specified list in a DataSource to which the report will bind.

Figure 31

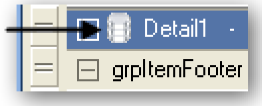
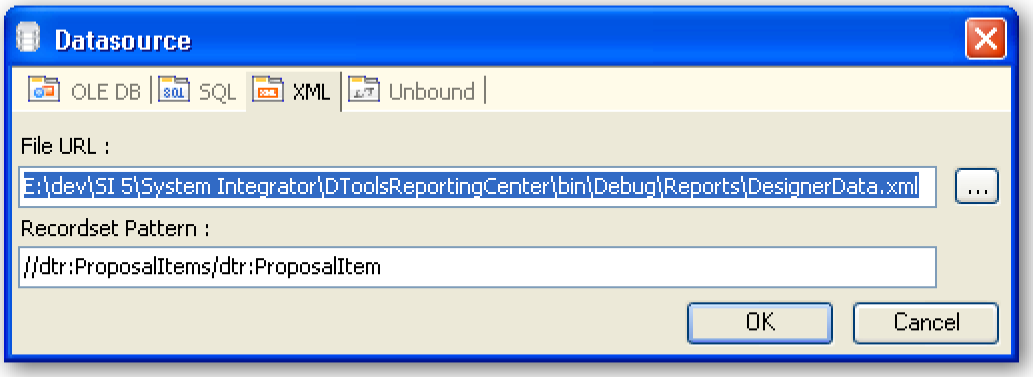


Figure 32

If a user is creating a custom report based on an existing report, there is no need to modify the

Datasource. By default, new custom reports NOT based on existing reports are set to iterate over the "Items" represented by the recordset pattern (XPath Expression): //dtr:Items/dtr:Item.



File URL : E:\dev\SI 5\System Integrator\DToolsReportingCenter\bin\Debug\Reports\DesignerData.xml

Recordset Pattern : //dtr:ProposalItems/dtr:ProposalItem

Figure 33

Following is a list of recordset patterns commonly used in DTools Reports:

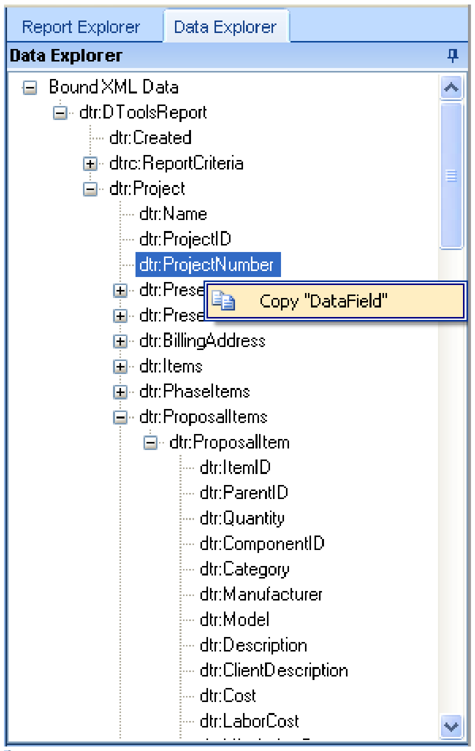
- **//dtr:Items/dtr:Item** – This pattern is used to iterate over each item in a project without regard to its existence in a package or its association as an accessory to another item. This is the "Engineering Bill of Materials". All physical items in a project in a list.
- **//dtr:ProposalItems/dtr:ProposalItem** – This pattern is used to iterate over each item in the "Sales Bill of Materials" and is the pattern used for the "Proposal" report and all of its variants. Top level items may represent Equipment items or Packages in a Project. Accessories to Equipment Items and the Items in a package are represented as child nodes to this node.
- **//dtr:PhaseItems/dtr:Item** – This pattern is used to iterate over items that must always be sorted by phase. A number of sub report rely on this section to render correctly.
- **//dtr:Project** – This pattern is used for reports that do not require the items to be iterated over. The Contract, Statement of Work and Cover Page reports are all examples of reports that do not iterate over items to render their data.

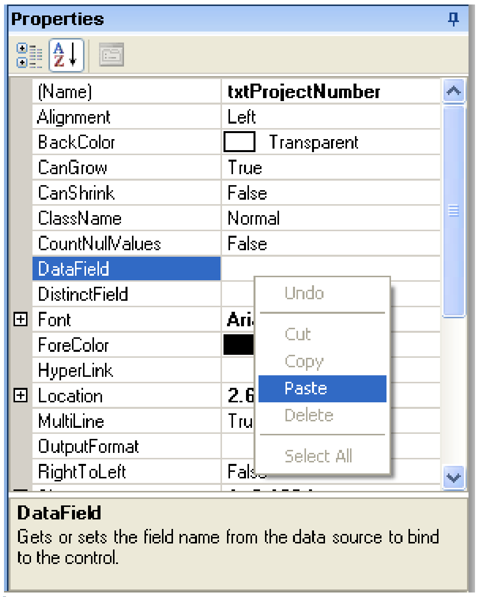
While these are the most common patterns used in the reports, any XML node can be used as the recordset pattern for a report. The "File URL" should not be modified when editing a Datasource in the Report Designer.

ADDING REPORT DESIGNER CONTROLS TO A REPORT

There are various types of controls that can be added to a report. Controls are added to a report by selecting the control in the Toolbox (Figure 29) and either dragging and dropping the control onto the Report Designer Canvas or placing your mouse on the design canvas, click and hold the left mouse and drag a rectangle where the control should be drawn. While details for these controls can be found on the Data Dynamics website http://www.datadynamics.com/Help/ARNET3/ActiveReports3_start.htm, a few key controls (and their properties) are worth highlighting here.

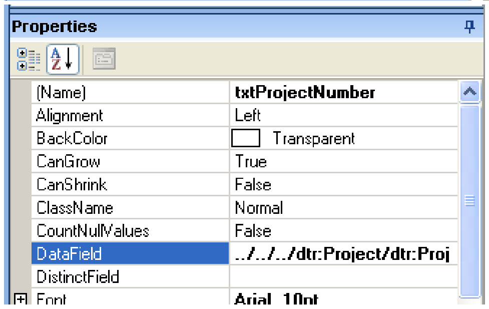
- **TextBox Control** – The Textbox control is the primary control used to represent data on a report.
    - **DataField Property** – The DataField property is set to the name of the data point in the data source the textbox should be bound to. In the case of SI5 reports, all reports are bound to an XML data source. This field should be set to a relative path based on the Recordset pattern of the Datasource. There are a number of ways to accomplish this. The first (and most painful) is to manually type the relative XPath into the DataField properties value. There are two easier methods:
        - If the user has added to Textbox from the Toolbar, they can use the "Data Explorer" Window to select the proper node in the XML data structure, right click on the node, select "Copy DataField" from the context menu, click on the DataField property in the Properties Window, right click and select "Paste".





(Name)	txtProjectNumber
Alignment	Left
BackColor	☐ Transparent
CanGrow	True
CanShrink	False
ClassName	Normal
CountNullValues	False
DataField	
DistinctField	
Font	Ari
ForeColor	■
HyperLink	
Location	2.6
MultiLine	Tru
OutputFormat	
RightToLeft	Fals

DataField
Gets or sets the field name from the data source to bind to the control.



(Name)	txtProjectNumber
Alignment	Left
BackColor	☐ Transparent
CanGrow	True
CanShrink	False
ClassName	Normal
CountNullValues	False
DataField	../../../dtr:Project/dtr:Proj
DistinctField	
Font	Arial, 10pt

Notice that the DataField property was set correctly to the relative path of the node in the recordset pattern.

        - The second way to correctly set this value is to simply select the proper node in the Data Explorer Window and drag and drop it onto the Report Design Canvas. This will automatically create a Textbox control on the design surface in the location where the node was dropped and set the DataField property of the newly created Textbox to the correct relative path.

    - **OutputFormat Property** – The OutputFormat property is used to apply a specific format to the value of the textbox. Common formats include formatting for currency and dates. In the existing reports if one sees "C" as the property of this value, it denotes that the value will be formatted as currency. While the format can also be set to "$0.00", using the letter "C" allows the formatting to adjust based on the regional settings of the computers operating system. If the user has the regional settings to English/US, then the currency will be formatted as dollars, but if they have the regional settings set to English/United Kingdom, the currency will be formatted automatically as British Pounds.
      The same concept applies to fields where the OutputFormat is set to "d". This automatically applies the computers short date format to the value. This accommodates the differences in countries where the date is written as day/month/year as opposed to the US formatting of month/day/year.

    - **SummaryFunc Property** – A list of functions that may be applied to the values of a textbox control for summing, counting, averaging, etc. values.

    - **SummaryRunning Property** – Value to determine whether summary values will be accumulated or reset for each level.

    - **SummaryType Property** – Sets the Type of summary to be performed on the values for this field. Valid options are (From the Data Dynamics documentation):

        - GrandTotal - Summary function for all records in the report.

        - PageTotal - Summary function for all records on a page.

        - SubTotal – Summary function for all records in a group level.

        - PageCount – Print the page count or page number. Use with "SummaryRunning" set to All to print the page number.

    - **SummaryGroup** – Name of the group used to reset the summary value when the textbox control is calculating subtotals. This property is only used when the SummaryType property is set to "SubTotal".

- **Label Control** – Used to create static text in a report.

- **RichTextBox Control** – Allows Rich text documents to be loaded into a report.

- **SubReport Control** – Links, loads, and executes a report as part of the main report. Subreport controls can be hard to see in the report designer as they are represented by a light gray box on the report design canvas. It can be easier to select sub reports in the report layout by using the "Report Explorer".

ADDING, EDITING, OR DELETING SCRIPT

Some reports require specific functionality that cannot be accomplished without the use of some custom script. Conditional behavior based on parameter values is a good example of something that requires custom scripting. To access the script for a report, select the "Script" tab at the bottom of the report designer window (Figure 34). Custom script can be written in either VB or C# (based on the "ScriptLanguage" property of the report). All standard reports that require script have the script written in VB.

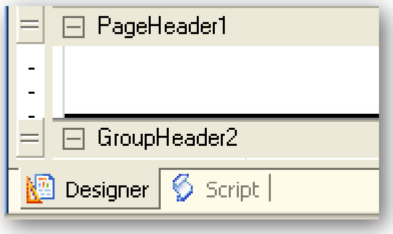
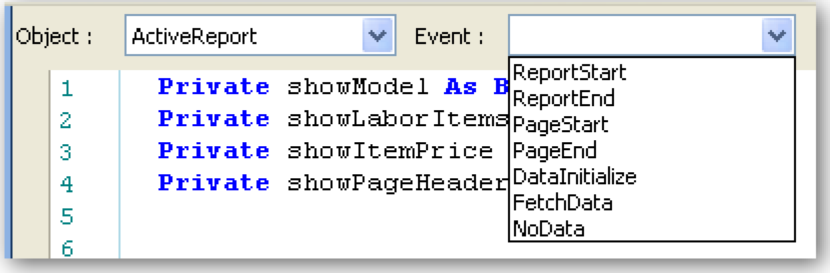


Figure 34

At the top of the script window, the user can select an object and event for which to write script (Figure 35). For specific information regarding the objects and available events please reference Active Reports Section Events.



```
Private showModel As B
Private showLaborItems
Private showItemPrice
Private showPageHeader
```

Figure 35

NOTES ON SCRIPTING

- Active Reports Scripting Overview
- Leverage the ReportUtilities Class

CHANGING THE REPORT INFORMATION

There may be times when the information that defines the report (Data set in the "New Report Wizard") needs to be modified for an existing custom report. This can be accessed via the main menu by selecting File > Edit DTools Report Information (Figure 36). This will reopen the Report Wizard for the currently loaded report.

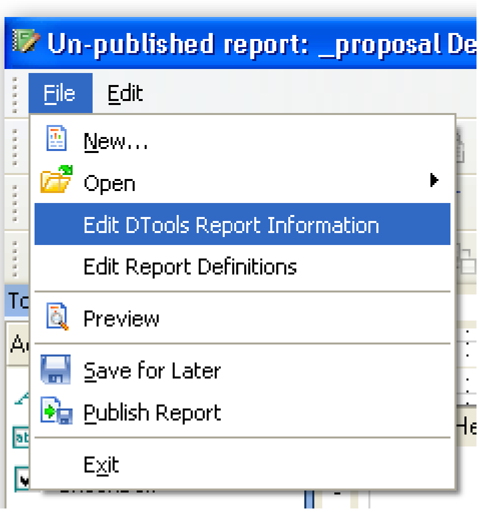
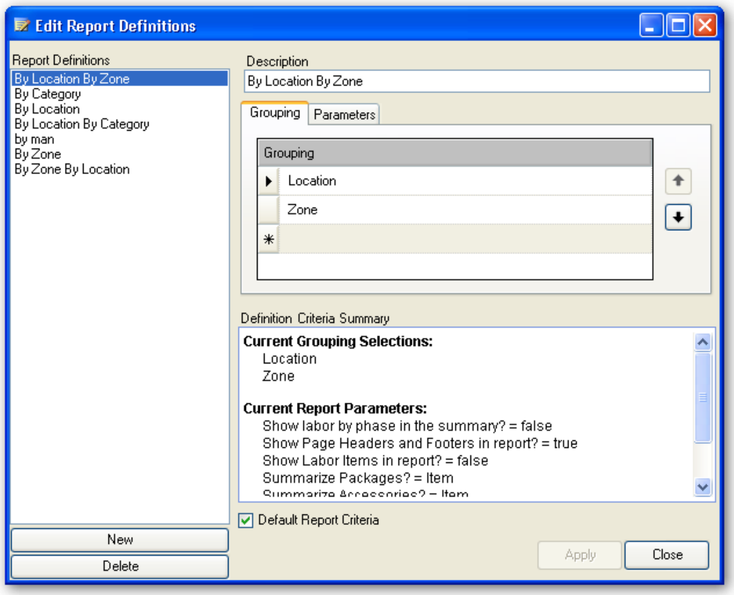


Figure 36

CREATING REPORT DEFINITIONS

When creating custom reports, users may wish to define report definitions for the report during the design phase. This can be especially handy if the user building the report needs to create multiple report definitions. This can be accessed via the main menu by selecting File > Edit Report Definitions (Figure 36). When a user is creating a report based on an existing report, any Report Definitions defined for the template report will be automatically added to the new report. The UI and functionality for this is very similar to the UI when a user is adding or editing Report Definitions in the Reporting Center (Figure 37). The only difference is that within this context, a use can add, edit, or delete multiple Report Definitions without closing the window.



Current Grouping Selections:
Location
Zone

Current Report Parameters:
Show labor by phase in the summary? = false
Show Page Headers and Footers in report? = true
Show Labor Items in report? = false
Summarize Packages? = Item
Summarize Accessories? = Item

Figure 37

SAVING AND PUBLISHING REPORTS

When a report is initially loaded, it is created as “Unpublished”. This allows users to close the Designer without having to finish designing a custom report. Once a report is complete, the user should select “Publish Report” from the toolbar or by selecting File > Publish Report (Figure 36).

When a user publishes a report the report is compiled and executed in the background. This is done to test for syntax errors in the reports script. If there are no compilation errors in the script, the user will be notified that the report has been published and the user will be prompted to either close the Designer and return to the Reporting Center or simply close the current report. If there are compilation errors in the script, the user will be presented with an error message. In many instances, this message will contain a line number where the problem in the script exists. These messages can be frustratingly vague however. Since these messages come from the Data Dynamics engine, DTools has no control on exactly how detailed (or not) these messages are. Chances are, if one is creating a report based on an existing report, they have deleted a control that has a script dependency.

There are many strategies for minimizing this type of issue. The following is a list of some options:

- Set the “Visible” property of unwanted controls to false instead of removing them.
- “Preview” the report after every few changes. If there any issues, the user will have a better sense of generally what may have caused the problem.
- Make a note of the name of the control being moved or deleted from the report and check the script for references to that control.

OPENING EXISTING REPORTS

Existing custom reports may be modified one of two ways:

- Via menus: File > Open > Open Existing Report (Figure 38):

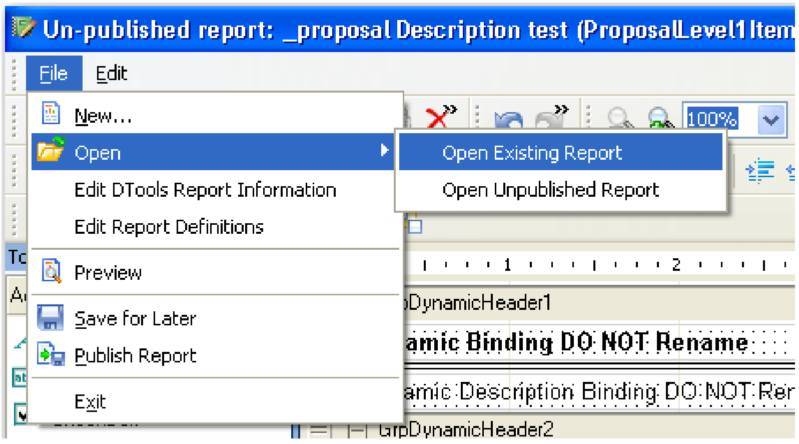

Un-published report: _proposal Description test (ProposalLevel1 Item
File
Edit
New...
Open
Edit DTools Report Information
Edit Report Definitions
Preview
Save for Later
Publish Report
Exit
Open Existing Report
Open Unpublished Report
100%
DynamicHeader1
amic Binding DO NOT Rename
amic Description Binding DO NOT Rer
GrpDynamicHeader2

Figure 38

- Via main Toolbar (Figure 39):

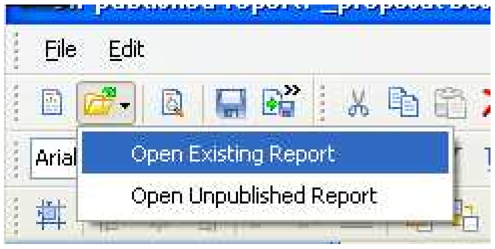
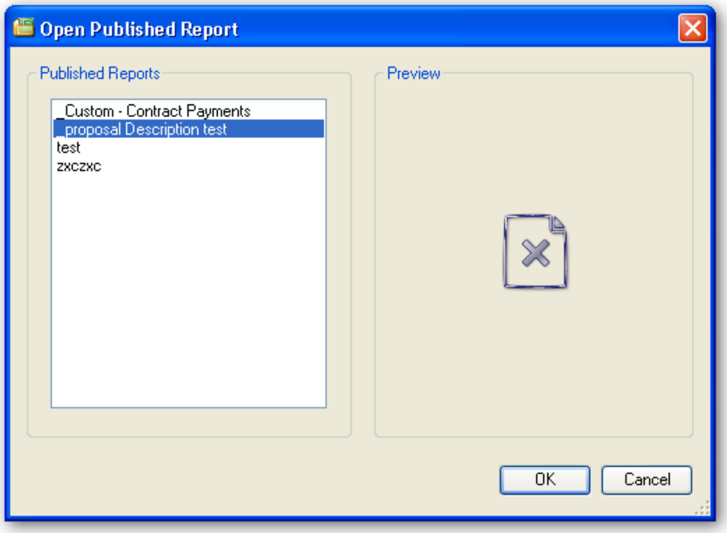


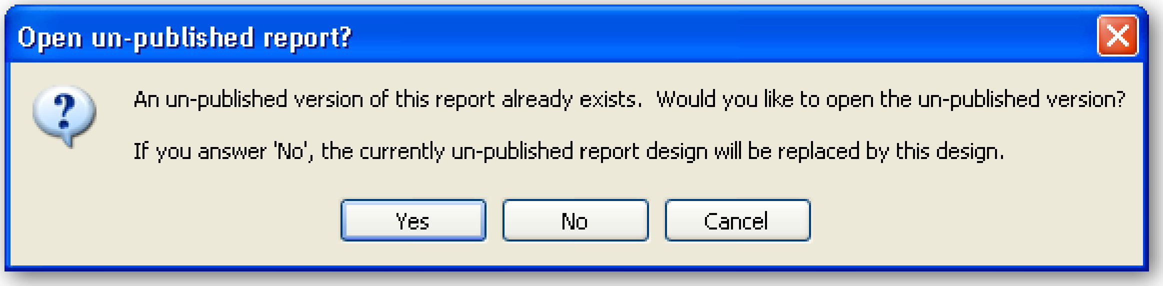
Figure 39

When the user opens executes one of the two means above, they are presented with a dialog to select an existing custom report (Figure 40). When the user selects the report and clicks “OK”, the report designer checks to see if there is an “Unpublished” version of the selected report currently saved. If so, the user will be prompted as to whether they want to load the currently unpublished version of the report or start a new version based on the currently published report (Figure 41). This check is to ensure that a user does not lose any changes they may have previously saved, but forgotten about. If the selects “Yes” at this prompt, the unpublished version will be loaded and the user will continue to modify that version. If the user selects “No”, the unpublished version will be replaced with a new unpublished version that is initially identical to the currently published version. Unpublished reports can be opened directly by selecting “Open Unpublished Report” instead of “Open Existing Report” from one of the two options above.



Custom - Contract Payments
proposal Description test
test
zxczxc

Figure 40



An un-published version of this report already exists. Would you like to open the un-published version?

If you answer 'No', the currently un-published report design will be replaced by this design.

Figure 41

GENERAL NOTES ON REGARDING THE REPORT DESIGNER

- A detailed explanation of all report controls and their properties are available here: Data Dynamics Online Help for Active Reports
- Reports only iterate over one section of the XML document (set as the recordset pattern of the Datasource). Having a report show repeating data over different sections of the XML requires the use of sub reports. If a user adds a textbox control and sets the DataField property of the control to a node that may be repeated in the XML, the report engine will return the value from the first node in the Node list for that section of the XML. In other words, if the report is iterating over items, and the user binds a text box to "dtr:CostDescription" node of the Misc. Cost section of the XML, since there can be more than one "dtr:MiscCosts" nodes in the XML, the value returned to the report will always be the value in the first node.

- Selecting an Item name in the "Report Explorer Window" will select that item on the Report Designer Canvas. This is helpful when a user is having a hard time locating a specific control on the canvas (sub reports are an excellent example of this), but knows its name and the section in which it exists (Figure 42). In the image below, the when the control name is selected in the Report Explorer (Right), the control is selected in designer (Left).

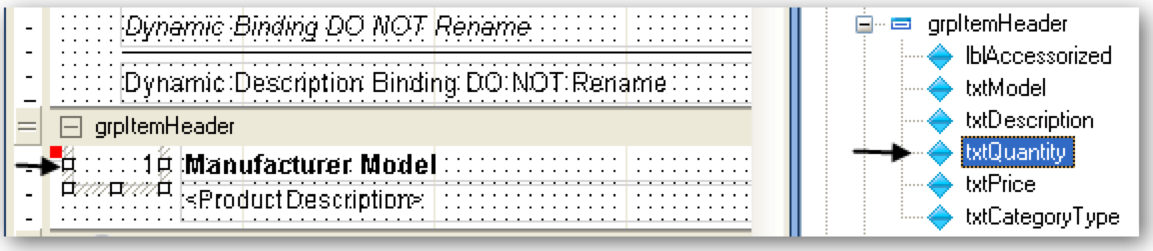
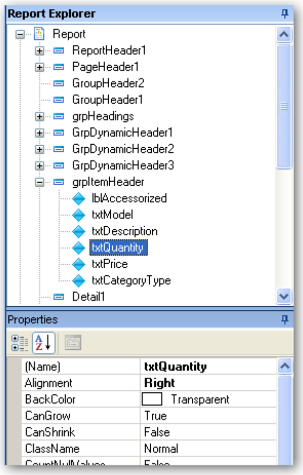


Figure 42

- Selecting an item name in the "Report Explorer Window" will set it to the selected item in the Properties Window (Figure 43).



(Name)	txtQuantity
Alignment	Right
BackColor	Transparent
CanGrow	True
CanShrink	False
ClassName	Normal

Figure 43

DYNAMIC BINDING DETAILS

If a report supports "above", the report merges the data defined in the Report Definition with the report layout. It binds values to sections of a report based on the names of the sections and the fields contained in the section. When a report supports "above", the reporting engine looks for one or more of the following sections to bind "group by" fields to. This allows a single report to support grouping in a number of different manners without requiring the specifically bound sections exist in the report prior to execution.

While the names of the dynamically bound sections and controls are rigidly set, the layout and formatting are completely flexible. Furthermore, other fields can be added to the sections and early bound to a DataField in the report data if desired.

THE SECTIONS MUST BE DEFINED AND NAMED AS FOLLOWS (FIGURE 44):

GrpDynamicHeader1 (GroupHeader object)

- Bindable fields are as follows:
    - txtDynamicHeader1 (Name of the field being bound) (Textbox object)
    - txtDynamicDesc1 (will bind description (location, zone) (Textbox object)

GrpDynamicHeader2 (GroupHeader object)

- Bindable fields are as follows
    - txtDynamicHeader2 (Name of the field being bound)
    - txtDynamicDesc2 (will bind description (location, zone)

GrpDynamicHeader3 (GroupHeader object)

- Bindable fields are as follows
    - txtDynamicHeader3 (Name of the field being bound)
    - txtDynamicDesc3 (will bind description (location, zone)

GrpDynamicFooter3 (GroupFooter object)

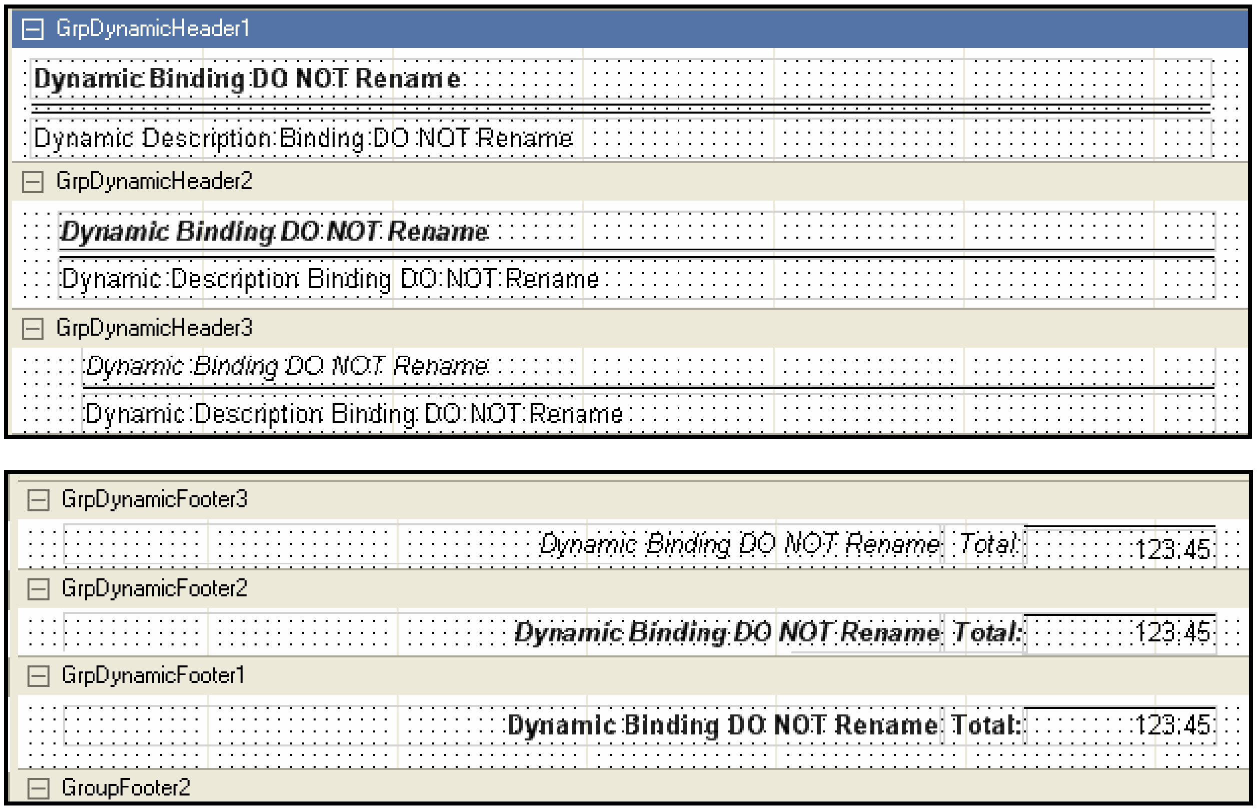
- Bindable fields are as follows
    - txtDynamicFooter3 (Name of the field being bound)

GrpDynamicFooter2 (GroupFooter object)

- Bindable fields are as follows
    - txtDynamicFooter2 (Name of the field being bound)

GrpDynamicFooter1 (GroupFooter object)

- Bindable fields are as follows
    - txtDynamicFooter1 (Name of the field being bound)


GrpDynamicHeader1

Dynamic Binding DO NOT Rename

Dynamic Description Binding DO NOT Rename

GrpDynamicHeader2

Dynamic Binding DO NOT Rename

Dynamic Description Binding DO NOT Rename

GrpDynamicHeader3

Dynamic Binding DO NOT Rename

Dynamic Description Binding DO NOT Rename

GrpDynamicFooter3

Dynamic Binding DO NOT Rename Total: 123.45

GrpDynamicFooter2

Dynamic Binding DO NOT Rename Total: 123.45

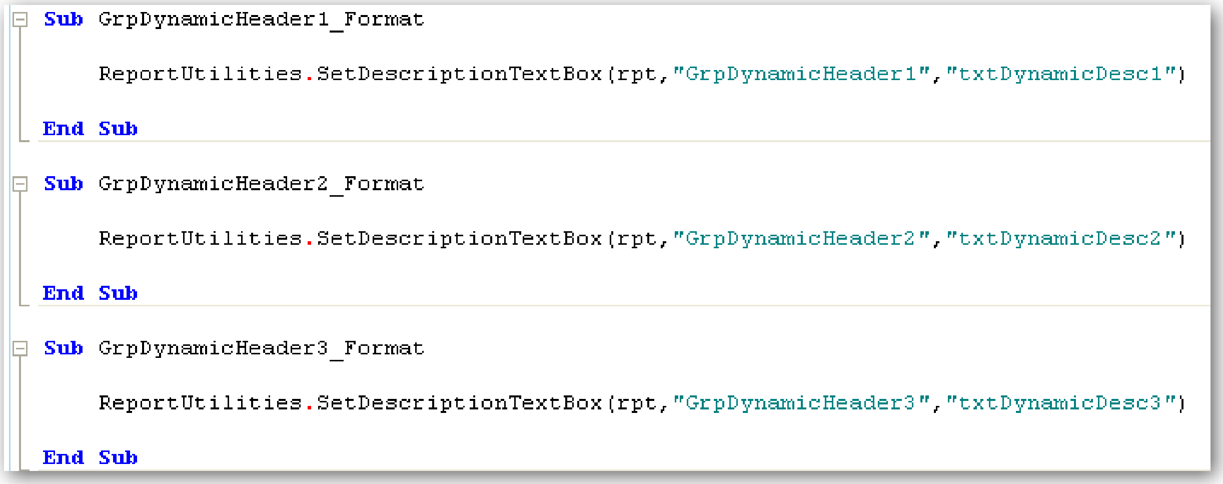
GrpDynamicFooter1

Dynamic Binding DO NOT Rename Total: 123.45

GroupFooter2

Figure 44

In addition to the sections being added, the following script events are required to support dynamic description binding for Locations and Zones:


```
Sub GrpDynamicHeader1_Format

    ReportUtilities.SetDescriptionTextBox(rpt,"GrpDynamicHeader1","txtDynamicDesc1")

End Sub

Sub GrpDynamicHeader2_Format

    ReportUtilities.SetDescriptionTextBox(rpt,"GrpDynamicHeader2","txtDynamicDesc2")

End Sub

Sub GrpDynamicHeader3_Format

    ReportUtilities.SetDescriptionTextBox(rpt,"GrpDynamicHeader3","txtDynamicDesc3")

End Sub
```

When creating a new report from scratch or basing one on an existing report that contains these fields, these sections and fields will be added automatically via the Report Wizard.

HOW ITEMS ARE GROUPED AND SORTED (SORTING STACK)

Instances of group sections in a report are created when the data of the field the GroupHeader is bound to changes. In order for group sections to be created properly, the data must be sorted based on the field to which the section is bound. In other words, if a report has a group section bound to “Location”, the data must be sorted by location for the report to create the correct sections.

In many instances, reports generated via the Reporting Center are grouped at multiple levels (Proposals in particular). In the case of reports that have multiple group sections. The data needs to be sorted appropriately. If a report has a group bound to Location, and another group bound to Zone, then the data needs to be sorted by location, then zone. This means that once all items are sorted by location, the items with matching locations are then sub sorted by zone.

In order to accomplish this, the Reporting Engine creates a Sort Stack. The sort stack is based on the merging of two pieces of information: The Report (as defined in the Report Wizard) and the Report Definition that is being run to create the report.

There are four components between the two objects above that determine how the items are ordered:

1. The Dynamic Groupings (defined in the Report Definition if the report supports dynamic grouping)
2. The Required Groupings (in the Report)
3. The Dynamic Sort Orders (in the Report Definition if the report supports Dynamic Sorting)
4. Required Sort Orders.

The data is sorted in the order above. Here’s how it works. Dynamic grouping implies that the report has sections that will be bound at runtime. These sections (between one and three levels) are the “top level” sections in the report so the data must be sorted by the values defined by the dynamic groupings. Required Groupings imply that there is a section of the report that is bound to a DataField at the time the report is designed. A good example of this is the section “grpItemHeader” in the Proposal reports. This section is bound to the ItemHash property at design time. Because the report section will be breaking on changes in the data, the grouping needs to be added at design time to the required groupings section. The Sort entries occur after the data has been sorted by the groupings.

There are a few things to keep in mind when considering the various levels to which the data is sorted.

- It only makes sense to sort the data to the level that the data is displayed. Some reports do not show details. With the Proposal, the most detailed level that the project data is displayed is when the items are grouped by ItemHash. This is a group section and the items are aggregated at this level. The individual records that make up this data is never displayed so allowing dynamic sorting or adding a required sort will have no impact on how the data is presented in the report.
- Groupings are nothing more than Ascending Sorts in the sorting stack. There have been questions about how one generates a Proposal with the items sorted by price in a descending order. This is a fairly complex question without a clear answer, but the short answer is that if one removes the Required Groupings and adds them to the Required Sort Orders, there is a bit more flexibility. See the thread on

our forums for detailed explanation: Sorting Products in a Report For an answer to the last post in the thread, see: Same Man & Model, Different Description

AGGREGATING ITEMS IN A PROPOSAL

As discussed in previous sections, the Proposal reports iterate over the "//dtr:ProposalItems/dtr:ProposalItem" section of the XML document. The Proposal reports aggregate like items in order to get "Quantity" counts. In order to aggregate items, the data must be sorted in a particular manner in order for like items to group (see previous section). Products in D-Tools are keyed by Manufacturer, Model, & Category. Items that have the same, Manufacturer, Model, and Category are considered the same. This works great in a limited scope, but SI allows users to accessorize items and modify the equipment items in packages. When either of these two scenarios occurs within the context of a project, the items can no longer be aggregated as they have become different items by virtue of the changes to their children.

The reporting engine handles this by generating an ItemHash value that represents the item. Each physical item has an ItemHash generated based on the items key. Furthermore, items with children (packages or accessorized equipment) include the ItemHash values of their child items when generating its ItemHash value. The net result is as follows:

A Project has 3 Items (assume category is the same):

Man: Sony, Model: 123, Price: $100

Man: Sony, Model: 123, Price: $100

Man: Sony, Model: 123, Price: $100

- Accessory: Man: Sony, Model: 456, Price: $20

When the items are aggregated in a Proposal, the results look as such (assume accessories are being hidden:

QTY	ITEM	PRICE
2	Sony 123	$200
1	Sony 123	$120 *

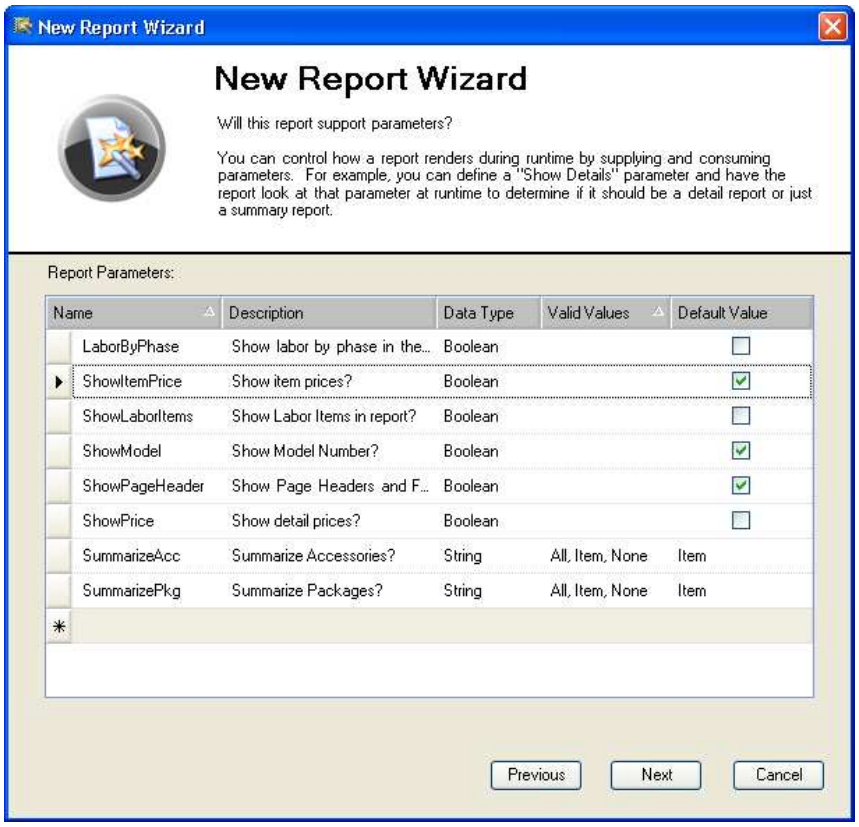
* Price includes accessories

The ItemHash for the first two items in the project is equal. The third item has a different ItemHash due to it having an accessory.

When one looks at the Report information in the Report Wizard, the user will see that the required groupings are Manufacturer, Model, and ItemHash. Even though the ItemHash is the field that the Proposals break on, sorting strictly by ItemHash will not keep the item data in any kind of alphabetical (or specific) order. By adding the Manufacturer and Model to the sort stack before ItemHash ensures that the Items in the example above, while not aggregated, are ordered next to each other or more generally, all items from a given manufacturer in a group are consecutively ordered.

CONSUMING PARAMETERS

When users are creating or editing a report, they have the option to add parameters to the report via the Report Wizard (Figure 45). These parameters act as a means for more granular control of the behavior of the report upon generation. All parameters added to a report will be added to any Report Definitions created for the report.



New Report Wizard

Will this report support parameters?

You can control how a report renders during runtime by supplying and consuming parameters. For example, you can define a "Show Details" parameter and have the report look at that parameter at runtime to determine if it should be a detail report or just a summary report.

Report Parameters:

Name	Description	Data Type	Valid Values	Default Value
LaborByPhase	Show labor by phase in the...	Boolean		☐
ShowItemPrice	Show item prices?	Boolean		☑
ShowLaborItems	Show Labor Items in report?	Boolean		☐
ShowModel	Show Model Number?	Boolean		☑
ShowPageHeader	Show Page Headers and F...	Boolean		☑
ShowPrice	Show detail prices?	Boolean		☐
SummarizeAcc	Summarize Accessories?	String	All, Item, None	Item
SummarizePkg	Summarize Packages?	String	All, Item, None	Item

Figure 45

Each parameter consists of the following components:

- **Name** – This is the key used to return the parameter value from the collection at runtime.
- **Description/Prompt** – This is the text that is displayed to the user when adding or editing a report definition.
- **Data Type** – This is the type of data the parameter value is. Parameters can be the following data types:

  - **String –** Text values.
  - **Integer –** Whole number values (i.e. 1, 2, 3, etc.)
  - **Double –** Floating point number values (i.e. 1.12, 1.345, 2.34, etc.)
  - **Boolean –** True or False.
- **Valid Values** – List of values from which to choose. This is available parameters with string, integer and double data types. In the Report Definition UI, parameters with valid values are represented as a drop down list of values.
- **Default Value** – The value the parameter is initially set to.

When a Report Definition is created, users can set the values of the parameters as they choose. Each Report Definition for a given report will have the same list of parameters, the values the parameters are set to, however, can be different for each report definition.

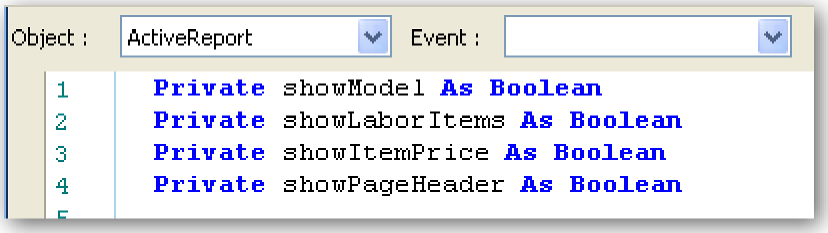
In the case of the Proposal, there is a "Show Model Number?" parameter. Users may create two report definitions for the report. One that has the "Show Model Number?" parameter set to 'True' and the other with it set to 'False'. Users can then simply run whichever Report Definition they choose depending on the need without having to open the Report Definition and change the value of the parameter each time.

While adding a parameter to the report is part of the equation, simply adding one does not have any effect on the behavior of the report. To do this, requires the user to write some script in the report. Below outlines this process (example in VB).

1. Create a variable in script to store the value of the parameter. This is done at the very top of the script window, outside of any functions or script event handlers (Figure 46). Declaring these variables outside of any methods gives them a global scope. This ensures that the values are available anywhere in the script without having to return the value everytime.

   The syntax is as follows:
   Dim **{variable name}** as **{type}**.

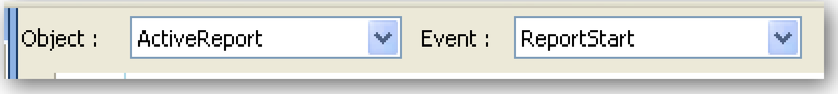
   Examples of variable declarations are as follows:
   ```
   Private boolean_Value as Boolean
   Private string_Value as String
   Private integer_Value as Integer
   Private double_Value as Double
   ```



```
Private showModel As Boolean
Private showLaborItems As Boolean
Private showItemPrice As Boolean
Private showPageHeader As Boolean
```

Figure 46

2. Select "ActiveReport" from the Object dropdown and "ReportStart" from the Event dropdown. This will create a script event handler for this event (if it does not exist) or place the cursors focus into this event handler (if it exists). Generally, parameter values should be returned in the "ReportStart" event to ensure the values are available throughout the complete life of the report generation process.



3. Return the value of the parameter via script and set the variable to the value. The syntax is as follows:

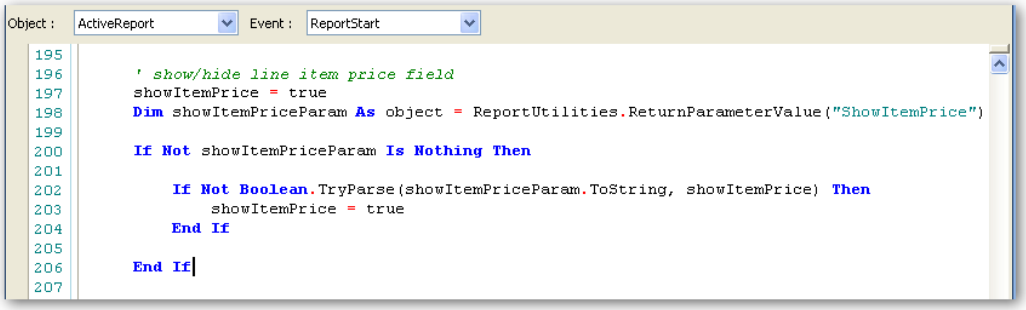
```
'Get the parameter
Dim param As object = ReportUtilities.ReturnParameterValue({parameter name})


'If nothing, set it to an empty string

If Not (param Is Nothing) Then

       {variable name} = ctype(param, {data type})

End If
```

In the above syntax, there are three tokenized sections:

{parameter name} – the name of defined parameter
{variable name} – name of the variable defined in step 1
{data type} – type of the parameter data (as defined in the Report Wizard).

The method "ReturnParameterValue" returns a value of type object (the base type for all other types). If the returned value is not null (Nothing in VB), the value is cast as the correct data type. The method "ReturnParameterValue" returns a null value if a parameter by the name passed into the method does not exist.

Below are code examples for returning parameters of each data type with inline comments. In each instance, the parameter value is initially returned as an Object. If the object is not null (Nothing in VB), conversion to the correct data type is attempted using the "TryParse" method on the data type. If the conversion fails, the variable defined for the parameter is set to a default value in script. This is done to ensure that the report executes properly by default (Figure 47).



```
    ' show/hide line item price field
    showItemPrice = true
    Dim showItemPriceParam As object = ReportUtilities.ReturnParameterValue("ShowItemPrice")

    If Not showItemPriceParam Is Nothing Then

        If Not Boolean.TryParse(showItemPriceParam.ToString, showItemPrice) Then
            showItemPrice = true
        End If

    End If
```

Figure 47

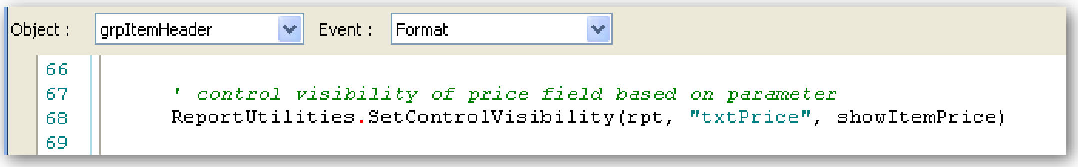
Much of the testing is in place simply from a precautionary standpoint. Given the ability for users to modify reports, it is possible that a user may delete a parameter in the Report Wizard, but leave the script. The testing for return values and parsing tests attempt to minimize the negative impact of a user deleting a parameter from the report, but leaving the supporting script in the report.

```
'RETURN a BOOLEAN parameter value

'Get the parameter value by passing the name of the parameter
Dim param As Object = ReportUtilities.ReturnParameterValue("boolean_Param")


'If the return value is not null (nothing), attempt to parse it as a boolean

'If the parsing fails, set it to a default value

If Not (param Is Nothing) Then

       If Not Boolean.TryParse(param.ToString, boolean_Value) then

              'Parsing failed.  Set it to a default value (1)

              boolean_Value = True

       End if

Else

'null value returned from the helper method, set default value


       boolean_Value = True


End If

'RETURN a String parameter value
```

```
'Get the parameter value by passing the name of the parameter
Dim param As Object = ReportUtilities.ReturnParameterValue("string_Param")


'If the return value is not null (nothing), call .ToString to get the value

If Not (param Is Nothing) Then

       string_Value = param.ToString()

Else

       string_Value = "default string value"

End If



'RETURN an INTEGER parameter value

'Get the parameter value by passing the name of the parameter
Dim param As Object = ReportUtilities.ReturnParameterValue("integer_Param")


'If the return value is not null (nothing), attempt to parse it as an integer

'If the parsing fails, set it to a default value

If Not (param Is Nothing) Then

       If Not Int32.TryParse(param.ToString, integer_variable) then

              'Parsing failed.  Set it to a default value (1)

              integer_Value = 1

       End If

Else

       'Null value returned from the helper method, set default value

       integer_Value = 1


End If



'RETURN a DOUBLE parameter value

'Get the parameter value by passing the name of the parameter
Dim param As Object = ReportUtilities.ReturnParameterValue("double_Param")
```

```
'If the return value is not null (nothing), attempt to parse it as an integer
'If the parsing fails, set it to a default value
If Not (param Is Nothing) Then
        If Not Double.TryParse(param.ToString, double_variable) then
                'Parsing failed.  Set it to a default value (1.23)
                double_variable = 1.23
        End If
Else
        'Null value returned from the helper method, set default value
        double_variable = 1.23


End If
```

Once the value from the parameter is returned, it can be used in script to control any type of behavior. In the image below (Figure 48), the visibility of the "txtPrice" Textbox is set based on the value of the "showItemPrice" parameter.


Object : grpItemHeader Event : Format

```
    ' control visibility of price field based on parameter
    ReportUtilities.SetControlVisibility(rpt, "txtPrice", showItemPrice)
```

Figure 48

STRUCTURE OF REPORTING FOLDER (BACKUP PURPOSES)

Reporting Data for each user is stored in the following folder:

XP - C:\Documents and Settings\{user name}\Application Data\D-Tools\SI5\Reports

Vista - C:\Users\{user name}\AppData\Roaming\D-Tools\SI5\Reports

- {user name} is the person currently logged into the machine.

The following files and folders should be copied when creating full backups:

- **ReportData.XML** – This file is located in the Reports folder. It is an XML document that contains all the information regarding available reports (standard, custom, and unpublished), categories, report definitions, and groups. A copy of this file (ReportData.bak) is created every time the Reporting Center is started as a restore point. If this file does not exist, it will be created the first time the Reporting Center is launched.

- **Report_Documents** – This folder contains the layout files (.RPX extensions) for custom reports.
- **Unpublished_Reports** – This folder contains the layout files (RPX extensions) for all unpublished custom reports.

Other key folders:

- **Project_Data** – This folder is where the XML files that the reports are generated against are stored. The contents of this folder are cleared every time the Reporting Center is launched. There is no reason to back this folder up.
- **Project_Delta** – This folder is where the XML files that represent "Delta" projects are created. When a user creates generates a delta between two revisions, the data file for the delta is generated in this folder. The contents of this folder are cleared every time the Reporting Center is launched. There is no reason to back this folder up.
- **Standard_Reports** – This folder contains the layout files (RPX extensions) for all standard reports. The contents of this folder are cleared and recreated every time the Reporting Center is launched. There is no reason to back this folder up.
- **Log** – Log files generated by the Reporting Center are stored in this folder. A log file is created every time the Reporting Center is launched. These files are saved for three days. Log files older than three days are deleted from this folder every time the Reporting Center is launched. There is no reason to back this folder up.