Quick Reports Reference Guide

Cavallo Support - 2026-08-20 - Reporting

Quick Reports is a module for SalesPad GP. Its primary purpose is to provide a quick mechanism to create and run simple reports based on SQL queries and user input. End users can view reports via the Quick Reports Viewer or via the Dashboard Viewer. At this time report writing must be performed with an external (text or XML) editor. The purpose of this Reference Guide is to provide report writers with the syntax (tags and attributes) that can be used to create a Quick Report.

See Also: Quick Report Designer

Table of Contents

- Report Tag
- Description Tag
- Search Tag
- Query Tag
- Column Tag
- Update Tag
- Script Tag

<report> Tag

The <report> tag is the required root element in a Quick Report.

Attributes

Attribute Name	Description	Value Options
Name	The name or short description of the Quick Report	(string value)
autoRun	A Boolean telling the Quick Reports viewer to automatically run the report when it is loaded.	True, false
DevExpress	The file name of DevExpress report filed tied to this Quick Report (usually *.repx). The file name should not include a path, and the file needs to be in the same folder as the Quick Report file.	(string value)
HideUndeclaredColumns	Set this to true to hide the columns from the query unless they are declared with a tag that has a Visible attribute set to true.	
AutoLinks	A Boolean telling the Quick Reports viewer to turn on/off the AutoLinks feature. SalesPad GP contains the following AutoLinks (the report should return the columns in the 2nd column to enable the desired AutoLink): Customer – Customer_Num Item – Item_Number SalesDocument – Source, Sales_Doc_Type, Sales_Doc_Num	
GroupFooterShowMode		
ShowAutoFilterRow		
ShowFooter		
UsePivotTable		True, false
ViewerName		PivotGridViewer

Child Tags

<description>, <search>, <query>, <column>, <compare>, <gridlayout>, <contextMenuItem>, <dropScript>

Example

<report name="All Quote Line Items And Quantities" autoRun="false" HideUndeclaredColumns="false" AutoLinks="false">

<description> Tag

The <description> tag is optional. It should contain a description of the report.

Attributes

None.

Child Tags

None.

Example

<description Text=”Text Description, this doesn’t show anywhere” />

<search> Tag

The <search> tag is used to describe a user input. The viewer will create an input box that the end user can populate and the query can use the input to filter data.

Attributes

Attribute Name	Description	Value Options
Name	The name/label of the input box that will be displayed to the user	(string value)
Column	The name of the column or parameter that will be used in the report query. The column name should be preceeded by an @ symbol if the query is a stored procedure.	(string value)
searchOp	The search operator used by Quick Reports to build the report query.	(any valid SQL operator like the following? like, =, <>, >,
Default Value	The default value for the input field	(string value)
PromptWidth		i.e. “200”
Type		DateTime,

Child Tags

None.

Example

<search name=”Item_Number” column=”Item_Number” searchOp=”like” />

<query> Tag

The <query> tag should surround the SQL query (or stored procedure) for the report.

Attributes

Attribute Name	Description	Value Options
addWhere	Tells Quick Reports to add (or not add) the “where” keyword to the query when running the SQL.	True, false

Inner Text

The SQL query can take the form of a raw query or a stored procedure call. A call to a stored procedure should be preceded by the "exec" keyword.

If you are using tags to get user input, Quick Reports will build a "where" clause or pass in parameters to a stored procedure. To manually specify the location of the "where" clause, you can use the key word “/*where*/".

Example

<report name="Customer List">
<search name="Customers Named Like:" column="CUSTNAME" searchOp="like"/>
<query addWhere="true">
<![CDATA[
select TOP 10 * into #tmp from RM00101 as Customers /*where*/ select * from #tmp where CreatedOn>’1/1/2007’
]]>
</query> </report>

Note: Because this query is contained in an XML file, appropriate XML conventions must be used; greater-than and less- than symbol escape codes must be used (>@lt;) unless you embed the query in a <![CDATA]]> section (see example above).

Child Tags

None.

Example

<query addWhere=”true”>
<! [CDATA[
select * from (
select sli.Sales_Doc_Num, sli.Item_Number, sli.Quantity from spv3SalesLineItem as sli

where sli.Sales_Doc_Type = ‘QUOTE’
)
as a
/*where*/
]]>
</query>

<column> Tag

The <column> tag is used to format a column returned by the query and displayed with the Quick Reports grid viewer.

Attributes

Attribute Name	Description	Value Options
Name	The name of the column returned by the query	(string value)

Attribute Name	Description	Value Options
Caption	The caption to display in the grid column header (the ^ symbol can be used to break a caption into multiple lines)	(string value)
BestFit	Tells the grid viewer to “best fit” the column caption and data	True, False
Sort	Tells the grid viewer to sort the column’s data	Ascending, Descending
DisplayFormat	Describes the format of the column’s data (.NET string format syntax)	(string value)
Band	The name (and caption) of the band that will be created and in which the column will appear in the grid viewer	(string value)
SummaryType	The summary type to display for a column in the footer of the grid. If the “Custom” value is used, you should also specify the CustomSummaryType attribute. DisplayFormat is a required attribute when you use the CustomSummaryType attribute	True, False
Visible	Can be used to hide a column (the column is visible by default and will not be hidden unless the attribute is specified with a “false” value)	True, False
CustomSummaryType	Describes the custom summary to be used in the grid footer for the column	Margin% (you must specify the SalesColumn and the CostColumn attributes)
SalesColumn	The name of the column that contains the sales number for use by the “Margin%” (string value – column name)	
CostColumn	The name of the column that contains the cost number for use by the “Margin%” (string value – column name)	
GroupSummaryType		Average, Sum, Min, Max, Count
Group		True, false
GroupDisplayFormat	Describes the format of the column’s data (.NET string format syntax)	(string value) *Note:* Only works with Average, Sum, Min, and Max as it should return integer value. Must use GroupSummaryType to access this item.
Condition		Equal, Less, Greater, true
ConditionVal1		(string value)
ConditionBackColor		.NET System.Color enum i.e. Red, Orange, Yellow, Blue, Green, White
ConditionToApplyRow		True
ConditionForeColor		.NET System.Color enum i.e. Red, Orange, Yellow, Blue, Green, White
ConditionFontStyle		Regular, Bold, Italic, Underline, Strikeout
ConditionFontStrikeout		True
ConditionFontSize		12.5, 14, etc.
ConditionFontName		Arial
PivotArea		RowArea, DataArea, FilterArea, ColumnArea
SalesDocXML		

Examples

```
<!------ Sets the color of the row without any conditions ------>
<column name="Sales_Doc_Num" Condition="true" ConditionApplyToRow="true"
ConditionFontName="Arial" ConditionFontSize="10.5" ConditionFontStrikeout="true"
ConditionBackColor="Blue" ConditionForeColor="White"/>
```

```
<!------ More Options for setting fonts on a column ------>
<column name="Item_Number" Condition="true" ConditionApplyToRow="true"
 ConditionFontName="Times New Roman" ConditionFontSize="16" ConditionFontStyle="Bold"
 ConditionBackColor="White" ConditionForeColor="Orange"/>
```

<!------ If Quantity Equals 1 then it will display the following condition colors ------>
<column name="Quantity" Condition="Equal" ConditionVal1="1" ConditionApplyToRow="true" ConditionBackColor="Black" ConditionForeColor="White" DisplayFormat="#,##0.#####"/>

<!------ If Quantity Is Greater Than 1 then it will display the following condition colors ------>
<column name="Quantity" Condition="Greater" ConditionVal1="1" ConditionApplyToRow="true" ConditionBackColor="White" ConditionForeColor="Black"/>

Child Tags

<update>

<update> Tag

The <update> tag is used to specify a stored procedure name (and parameters) that will be called after a change is made to data in the Quick Report (in the tag's parent column). Currently, only a stored procedure may be specified, and the procedure name and parameters must be enclosed with the corresponding child tags (see below).

Attributes

Attribute Name	Description	Value Options
ReturnVal	The name of the column to update if a value is returned from the stored procedure.	

Child Tags

None.

<script> Tag

The <script> allows a c# script to be fired during certain actions such as when the script is run or a keyboard shortcut is used.

Note: In the example blow, P1 is the name of the grid view. The OnRunScript tag allows you to call the grid view methods to update it.

Example 1

<OnRunScript>

p1.ExpandAllGroups
(); p1.CollapseAllGroups();

</OnRunScript>

Example 2

<!--
 Trigger: Right-Click -> "New" -> Opens New CRM Task Window

 Available Fields:
 - val_Assigned_To

```
  - val_Changed_By
  - val_Changed_On
  - val_Completed_On
  - val_Contact_Name
  - val_Created_By
  - val_Created_On
  - val_Customer_Name
  - val_Customer_Num
  - val_Description
  - val_Duration
  - val_Ends_On
  - val_Event_ID
  - val_Is_Complete
  - val_Remind
  - val_Remind_On
  - val_Reminder_Enabled
  - val_Sales_Person_ID
  - val_Starts_On
  - val_Status
  - val_Subject
  - val_Type
  - val_UserField
-->
<contextMenuItem text="New CRM Event...">
    SalesPad.Bus.CRMEvent crmEvent = new SalesPad.Bus.CRMEvent();

    // Task Details
    crmEvent.val_Subject = "Call List Item - From QuickReport";
    crmEvent.val_UserField = "Call List Created - CRM Item To Do";
    crmEvent.val_Description = "SampleNote";

    // Customer Information
    crmEvent.val_Customer_Num = "AARONFIT0001";
    crmEvent.val_Customer_Name = "Aaron Fitz Electrical";

    // Reminders
    crmEvent.val_Reminder_Enabled = true;
    crmEvent.val_Remind = "5 Minutes Before";

    // Scheduling
    System.DateTime today = System.DateTime.Now;
    System.DateTime answer = today.AddDays(30);
    crmEvent.val_Starts_On = answer;
    crmEvent.val_Ends_On = answer.AddHours(1);

    // Audit Info
    crmEvent.val_Created_By = "STEVE";
    crmEvent.val_Created_On = System.DateTime.Now;

    // Launch Window
    SalesPad.Args.CRMEventEntryArgs args = new SalesPad.Args.CRMEventEntryArgs(crmEvent, null);
    System.Framework.Controller.StartPlugin(args);
</contextMenuItem>
```

Example 3

```
<OnRunScript>
  System.Data.DataView dv = new System.Data.DataView(p1.GridControl.DataSource as
System.Data.DataTable);
  dv.RowFilter = "Created_By = '" + System.Framework.Controller.Credentials.UserName + "'";
  //System.Windows.Forms.MessageBox.Show(dv.RowFilter); p1.GridControl.DataSource = dv;
</OnRunScript>
```

Note: This script is used to only display results where the “Created By” name is the same as the “Username”

currently logged into SalesPad.